VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2019 Bc. Dominik Majer

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

FIRMWARE PRO ŘÍZENÍ SYSTÉMU KOMUNIKACE PO SILNOPROUDÉM VEDENÍ
FIRMWARE FOR POWER LINE COMMUNICAITON SYSTEM

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR
Bc. Dominik Majer

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Ján Sláčik

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Telekomunikační a informační technika**
Ústav telekomunikací

Student: Bc. Dominik Majer ***ID:*** 164761
Ročník: 2 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Firmware pro řízení systému komunikace po silnoproudém vedení

POKYNY PRO VYPRACOVÁNÍ:

Cílem diplomové práce je vytvořit firmware mikrokontroléru pro obsluhu SoC ST7580. Firmware musí umožňovat přenos dat (vývoj vlastních protokolů) po silnoproudém vedení (PLC - Power Line Communication), konfigurovatelnost PLC systému a umožňovat komunikaci s nadřazeným systémem (platformy PC/web). Vytvořený firmware bude implementován do dodaných testovacích zařízení.
Výstupem práce bude vytvoření plně funkčního firmware pro obsluhu SoC ST7580, ověření komunikace po silnoproudém vedení a její konfiguraci a řízení v rámci nadřazeného systému.

DOPORUČENÁ LITERATURA:

[1] BERGER, Lars Torsten, Andreas SCHWAGER a J. Joaquín ESCUDERO-GARZÁS. Power Line Communications for Smart Grid Applications. Journal of Electrical and Computer Engineering [online]. 2013, 2013, 1-16 [cit. 2018-09-16]. DOI: 10.1155/2013/712376. ISSN 2090-0147. Dostupné z: http://www.hindawi.com/journals/jece/2013/712376/

[2] GALLI, Stefano, Anna SCAGLIONE a Zhifang WANG. 2011. For the Grid and Through the Grid: The Role of

Power Line Communications in the Smart Grid. Proceedings of the IEEE. 99(6): 998-1027. ISSN 0018-9219.

Dostupné z: https://ieeexplore.ieee.org/document/5768099/

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 16.5.2019

Vedoucí práce: Ing. Ján Sláčik
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

UPOZORNĚNÍ:
Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

ABSTRAKT

V diplomové práci je rozebrána problematika komunikace po silnoproudém vedení a návrh síťové architektury s cílem vytvořit vlastní návrh a realizaci pro chytrý rodinný dům. Na základě vytvořeného konceptu jsou navrženy komunikační protokoly pro systém. Dále je provedena implementace návrhů, jejich testování a ladění v laboratorních podmínkách v reálných zařízeních. Shrnutí dosažených výsledků je podloženo ukázkami reálného provozu.

KLÍČOVÁ SLOVA

PLC, ST7580, C, C#, SQL, silové vedení, komunikace po silovém vedení, protokol

ABSTRACT

In this diploma thesis it was introduced the issue of power line communication and network architecture design with goal to create own concept of smart house system. Consequently, the necessary communication protocols are designed and described based on the concept. Further is described the implementation of the designs and testing of the realized system under laboratory conditions using the real communication devices. Summary of the achieved results is supported by demonstration of real functionality.

KEYWORDS

PLC, ST7580, C, C#, SQL, power line, power line communication, protocol

MAJER, Dominik *Firmware pro řízení systému komunikace po silnoproudém vedení*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2019. 103 s. Vedoucí práce byl Ing. Ján Sláčik

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Firmware pro řízení systému komunikace po silnoproudém vedení“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Nejprve bych rád poděkoval Bohu, své snoubence a nejbližší rodině za podporu a pomoc při řešení této práce zejména v situacích, kdy jsem se potýkal s následky onkologické léčby.
V neposlední řadě bych rád poděkoval vedoucímu diplomové práce panu Ing. Jánu Sláčikovi za odborné vedení, konzultace, trpělivost a zejména za velmi aktivní přístup, kterým mne posouval neustále dopředu.

Brno ....
(podpis autora)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této diplomové práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno ....

(podpis autora)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

V diplomové práci je rozebrána problematika komunikace po silnoproudém vedení – aneb zkráceně PLC (Power Line Communication). Tato technologie je světu známá již více než jedno století. Jak je uvedeno v [1] první zmínka o PLC komunikaci sahá až do 40. let 19. století a šlo o přenos velmi jednoduchých signálů. V roce 1897 se objevuje první patent spojený s touto technologií registrovaný ve Spojeném království [2]. Rozšíření této technologie proběhlo ve 20. až 30. letech 20. století a našla uplatnění jako jednoduchý nástroj pro komunikaci v případě selhání části distribuční sítě elektrické energie mezi jednotlivými distribučními stanicemi (elektrárny, trafo stanice), aby se minimalizovaly důsledky výpadku [1], [3]. Vzhledem k technickým možnostem šlo pouze o úzkopásmovou komunikaci. Komunikační pásmo bylo v řádu desítek až stovek kilohertzů. Tato myšlenka komunikace mezi distribučními stanicemi elektrické energie se používá dodnes. Na přelomu 30. a 40. let 20. století našla uplatnění i v domácnostech a šlo např. o dětské chůvičky [1]. Dle [4] se kolem roku 1950 tato technologie začala používat pro ovládání pouličního osvětlení a v 70. a 80. letech 20. století se začal již zkoumat potenciální duplexní přenos dat přes silové vedení [1]. První pokusy o přenos probíhaly řádově v pásmu stovek kilohertzů, respektive 5–500 kHz. Jak je uvedeno v [5] na konci 90. let se začal vývoj urychlovat spolu s nástupem rozmachu internetu, jak ho dnes známe. V dnešní době je k dispozici velké portfolium PLC modulů, které jsou určeny pro LAN (Local Area Network) aplikace s propustností v řádu stovek Mbit/s.

Práce se zabývá rozborem řešení a návrhem síťové architektury za účelem vytvoření autonomní PLC sítě, která je monitorována z externího systému; konkrétně pomocí webového rozhraní. Nejprve je vytvořeno programové vybavení pro zvolená zařízení. Nejedná se o velkoobjemovou datovou komunikaci, nýbrž o senzorickou a monitorovací síť, která přenáší informace mezi jednotlivými bránami a senzory. Dále je navržen způsob monitorování z nadřazeného systému (webový server). V práci jsou nejprve rozebrány jednotlivé bloky komunikačního řetězce, následně navrženy komunikační protokoly a nakonec na základě návrhů napsán výsledný kód. Ten je prakticky implementován a otestován. Dosažené či nedosažené výsledky jsou závěrem diskutovány. Nutno zmínit, že tato síť svou podstatou má prvky sítí IoT (Internet-of-Things). Pod tímto pojmem si lze představit monitorovací a signalizační sítě např. v chytrém domě, kde účastníky komunikace jsou např. termostat ovládající klimatizaci/topení, bezpečnostní senzory či i lednice. Základním prvkem vytvořené sítě je modul, který se připojí na silové vedení a představuje právě jednoho účastníka komunikace (konkrétní typ modulu je určen následným připojením přídavného modulu což může být právě např. zmiňovaný termostat). V dnešní době je již pro koncového zákazníka k dispozici široké portfolio koncových modulů určených k zapojení do standardní zásuvky. Zde je ale řešen nový přístup k problematice z hlediska vlastního návrhu síťové architektury, komunikačních protokolů a programové výbavy.

1 POUŽITÉ TECHNOLOGIE

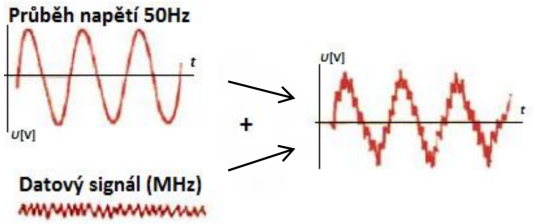
V této kapitole je nejprve popsán princip fungování PLC technologie. Nejprve je popsán samotný princip PLC a následně jsou rozebrány druhy PLC technologií, jejich příslušné normy a státní regulační orgány. Ve druhé části jsou rozebrány fyzické (HW; hardware) součásti použité pro vývoj.

1.1 PLC technologie

Obecný princip lze popsat na příkladu klasického rodinného domu. Ten je uvažován se standardně provedenou elektrickou instalací (členění na okruhy zásuvkové, světelné, které jsou vyvedeny v domovním rozvaděči). Z elektrického hlediska jsou všechny obvody propojeny[1] (sbíhají se v hlavním domovním rozváděči) a na stejném potenciálu (230 V / 50 Hz). Pokud se PLC zařízení připojí v jakémkoli konkrétním místě v domě na vedení, tak v jakémkoli libovolném místě v domě může jiné PLC zařízení odebírat datový PLC signál. Toto platí pro každou fázi. Problém nastává, když se komunikuje mezi jednotlivými fázemi [5]. Komunikace sice stále probíhá, ale dochází k útlumu signálu až o 20 dB. To lze řešit zavedením speciálním mezičlánkem, který do určité míry propojí příslušné fáze a sníží útlum [5].

Nyní je objasněn princip, jak se modulovaný nosný signál poslán po silnoproudém vedení. Vychází se z obr. č. 1, kde je v levém horním grafu je vykreslený signál na silnoproudém vedení. Na levé straně ve spodní straně obrázku se nachází signál s amplitudou několikanásobně menší, ale s několikanásobně vyšším kmitočtem. Poslední průběh je právě výsledný průběh, který vznikne po sečtení silového signálu se signálem datovým. V přijímači jsou následně nízkofrekvenční složky signálu odfiltrovány, takže zůstanou jen vysokofrekvenční složky, kde jsou namodulována samotná data. Následně se na základě použité frekvence nosného signálu rozlišuje PLC komunikace [5]:

- **Úzkopásmová** – od 3 kHz do 500 kHz (stovky kbit/s)
- **Širokopásmová** – od 1,8 MHz do 250 MHz (stovky Mbit/s)



Obr. 1: Grafická demonstrace principu PLC, převzato z [5], strana 10.

[1] I když je každý okruh samostatně jištěn, tak pokud jistič není v poruchovém stavu, nepředstavuje překážku ve spojení mezi jednotlivými okruhy.

1.1.1 Úzkopásmová PLC

Nosný signál se pohybuje ve frekvenčním rozsahu od 3 kHz do 500 kHz. Z toho vyplývá omezená šířka pásma a tudíž i omezená rychlost přenosu dat. Proto úzkopásmová PLC našla uplatnění především tam, kde se nevyžaduje přenos velkého objemu dat, ale spíše jen telemetrických či signálních zpráv.

V tab. č. 1 jsou uvedeny vybrané světové regiony, jejich regulační orgány pro úzkopásmovou PLC technologii a využívaná pásma. Pro potřeby práce je použita evropská norma, která je definována standardem CENELEC EN 50065 [2]. Z tab. č. 2 lze vyčíst, že norma definuje použití v pásmu 9–148,5 kHz. Toto pásmo je dále rozčleněno na pět podkategorií, z nichž každé má jiné využití. Níže jsou jednotlivé kategorie na základě informací [6] stručně popsány.

Tab. 1: Přehled nasazení úzkopásmové PLC ve vybraných regionech [6].

Region	**Regulační orgán**	**Pásmo [kHz]**	**Poznámka**
Evropa	CELENEC	3–95	pásmo A
		95–125	pásmo B
		125–140	pásmo C
		140–148,5	pásmo D
Japonsko	ARIB	10–450	—
Čína	EPRI	3–90	Neregulováno
		3–500	—
USA	FCC	10–490	—

Tab. 2: Evropská norma CENELEC EN 500653 [6].

Pásmo	**Šířka pásma [kHz]**	**Maximální amplituda signálu [V]**
—	3–9	—
A	9–95	1–5
B	95–125	1,2
C	125–140	1,2
D	140–148,5	1,2

Pásmo A Je určeno výhradně dodavatelům elektrické energie, ale na základě jejich souhlasu je dovoleno, aby jej využívali i odběratelé. Od odběratelů lze např. odečítat hodnoty elektroměru. Distributoři toto pásmo využívají pro energetický dohled a komunikaci mezi elektrárnami.

Pásmo B Slouží pouze pro odběratele elektrické energie a nevyžaduje protokol přístupu, který je definován normou. Pásmo se používá pro ovládání běžných spotřebitelských koncových zařízení připojených na silové vedení (osvětlení, topení, klimatizace), dále na řízení dalších systémů v domě (okna, dveře) a nakonec i pro propojení menších senzorů či bezpečnostních prvků. To znamená, že pásmo B se nasazuje v tzv. chytrých domech.

Pásmo C Aplikace jsou stejné jako u pásma B. Je též určeno jen pro odběratele elektrické energie. Jediný, ale velmi důležitý rozdíl je, že je vyžadován protokol o přistoupení k dohodě dle ČSN EN 50065.

Pásmo D Taktéž je určeno pouze pro odběratele, ale je zde vyžadován protokol přístupu dle odpovídající normy.

1.1.2 Širokopásmová PLC

Širokopásmová komunikace naopak využívá frekvenční pásma v jednotkách až desítkách MHz. Proto na rozdíl od úzkopásmové se vyznačuje mnohem vyšší přenosovou rychlostí [5]. To naopak za cenu zkráceného dosahu. V rodinném domě se nejedná o problém nijak zásadní, zatímco vyšší rychlosti přenosu jsou žádány. Pokud je cílená aplikace ve větší kancelářské budově, kde je požadováno vysokorychlostní připojení k síti, je nutné použít opakovačů. Proto je širokopásmové připojení PLC jen velmi omezeně rozšířeno ve venkovním použití a omezuje se především na budovy [7].

1.1.3 Výhody a nevýhody využití PLC

Vzhledem k existenci jiných technologií jako WiFi, LTE či Ethernet, je nutné se zbývat také výhodami/nevýhodami PLC technologie, díky kterým je pak PLC technologie uvažována, či nikoliv. V článku [8] jsou klady a zápory PLC rozebrány a tato tvrzení lze podložit i další prací [1]. Níže jsou shrnuty jednotlivé výhody/nevýhody, důvody, proč se k použití PLC technologie rozhodnout, kdy se technologie PLC vyplatí a kdy naopak ne. Toto shrnutí je provedeno na základě informací z [8] a [1].

Výhody, přednosti a motivace pro využití

- Nízké náklady z pohledu přenosového média – kabeláž je a bude na místě nezávisle na tom, zda chceme PLC využít či nikoli (již existující infrastruktura sítě).
- Praktická propustnost v praxi na úrovni aplikační vrstvy kolem 200 Mbit/s (širokopásmová PLC).
- Teoretická propustnost na úrovni fyzické vrstvy dnes běžně používaných standardů kolem 500 Mbit/s (širokopásmová PLC).
- Svoboda zapojení – technologie není v praktickém nasazení příliš regulována (až např. na podmínky tónové masky), jelikož infrastruktura sítě je již na místě a bude tam i když se nebude PLC využívat.
- Kamkoli je vedeno silové vedení, tam je nasazení PLC možné.

Nevýhody a omezení

- Nejednotnost napětí či frekvence v distribuční soustavě napříč státy.
- Problém při komunikaci mezi okruhy/fázemi (interference/útlum)
- Při širokopásmové přenosu nastává problém, že silové vodiče jsou elektromagneticky nestíněné a chovají se jako anténa – dochází k elektromagnetické interferenci s dalšími komunikačními systémy, či se na vedení kumulují další signály (např. rušení z elektrických motorů).
- Nutné držet vysílací výkon PLC modemů podle regulace.
- PLC modemy musejí být připraveny na časté výkyvy síťového napětí.
- Impedanční přizpůsobení sítě.
- Odrazy na vedení – vedení není přizpůsobeno a ani se to nepředpokládá.

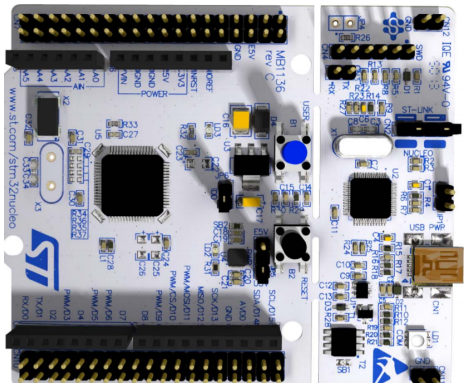
1.2 Testovací HW součásti

Cílem práce je především návrh architektury komunikace a z toho vyplývající tvorba programového vybavení pro dvě vývojové desky (kity), které spolu tvoří koncový komunikační PLC modul, respektive modem. Tyto dvě vývojové desky jsou popsány níže.

1.2.1 Vývojový kit STM32F401

Jedná se o vývojovou sadu použitou pro řízení koncového PLC modulu. Ilustračně je zobrazena na obr. č. 2. Popis uvedeného kitu vychází z informací udávaných výrobcem na svých oficiálních webových stránkách [10]. Dále je tento kit označován jako kontrolér či MCU (Micro Controller Unit).

Kód je vytvořen v jazyce C/C++ v Keil IDE (demo verze). Pro nastavení I/O periferií a inicializaci zdrojů desky se využívá program STM32CubeMX, který slouží pro intuitivní grafické pojmenování / nastavení pinů / časovačů / periferií na desce. Program CubeMX následně vygeneruje inicializační kód pro danou desku, tzn. není potřeba psát konfiguraci ručně. To je velká výhoda, jelikož by nejprve bylo nutné nastudovat rozložení registrů a jejich naplnění za účelem správné funkce periferií.



Obr. 2: Ilustrační fotografie desky STM32F401 Nucleo-64.

Jádro desky STM32F401 Jádrem desky je ARM (Advanced RISC Machine) procesor harvardské architektury Cortex-Mx[2]. Jedná se o skupinu 32bitových ARM procesorů licencována firmou Arm Holdings. Všechny procesory z této rodiny podporují funkcionalitu FPU (Floating Point Unit) nebo-li modul pro operace s plovoucí desetinou čárkou. Použité jádro Cortex-M4 je jádro Cortex-M3 doplněné o FPU a DSP (Digital Signal Processor). Všechny klíčové vlastnosti jsou shrnuty v tab. č. 3.

Tab. 3: Základní parametry desky STM32F401 [10].

Paměť	Flash	512 kB	—
	SRAM	96 kB	—
Komunikační rozhraní	I2C	3×	—
	USART	3×	2× až 10,5 Mbit/s 1× až 5,25 Mbit/s
	SPI	4×	až 42 Mbit/s
	USB	1×	verze 2.0, OTG
Porty	celkem 81 portů s možností externího přerušení		
	celkem 78 portů uzpůsobeno pro 42 MHz		
Časovače	16bitový	5×	—
	16bitový	1×	pro PWM za účelem řízení motorů
	32bitový	2×	až 84 MHz
Krystalové oscilátory	32 kHz	1×	—
	26 MHz	4×	—
Interní RC oscilátory	32 kHz	1×	—
	16 MHz	1×	—
- Podpora DMA technologie			
- Speciální modul pro kalkulaci CRC			
- 1× 16kanálový A/D převodník			
- Rozmezí pracovního napětí 1,7 V až 3,6 V			

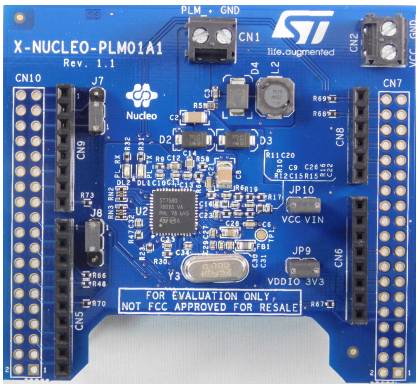
1.2.2 Rozšiřující deska X-NUCLEO-PLM01A1

Jedná se o tzv. shield pro vývojové desky/kity (zmíněná STM32F401) a její popis též vychází z oficiální zdroje od výrobce dostupného z [11].

Tato deska je stavěná zejména pro vývojové kity od STM. Je ale uzpůsobená i pro určité druhy např. Arduino vývojových desek. Jádrem je SoC ST7580 (zkratka z "System on Chip"). Označení SoC značí, že čip obsahuje předem definovaný firmware, který určuje funkci čipu.

[2]Za písmeno "x" se dosadí konkrétní modelová řada.

PLC přijímač a vysílač jsou optimalizovány pro standard CELENEC pásmo B, ale podporuje i pásma A, C, D. Deska neobsahuje člen pro připojení na silové vedení (tzv. vazební člen) a je nutné připojit např. článek opět od firmy STMicroelectronic s označením STEVAL-XPLM01CPL [11]. Ilustrační obrázek X-NUCLEO-PLM01A1 viz obr. č. 3.



Obr. 3: Ilustrační fotografie desky X-NUCLEO-PLM01A1.

Čip ST7580

Na základě vlastností udávaných výrobcem [12] se jedná o SoC, proto je nutné přizpůsobit komunikaci dle výrobcem definovaného protokolu. Čip poskytuje kompletní služby fyzické vrstvy PLC komunikace a několik služeb vrstvy linkové. Hlavními aplikacemi pro čip jsou přenos telemetrie či monitorování, vzdálené ovládání, kontrola a režie ovládání pro chytré domy. Může pracovat se dvěma typy klíčování – PSK a FSK a jejich některými modifikacemi. Dále umožňuje práci s konvolučním kódováním (obsahuje zvláštní modul určený pro tyto účely). Mezi další služby linkové vrstvy patří detekce chyb v přijatých rámcích a šifrování pomocí AES.

Klíčové vlastnosti čipu ST7580

- Podpora úzkopásmové PLC komunikace ve frekvenčním rozsahu 0–255 kHz,
- podporované klíčovací techniky – BFSK, BPSK, QPSK, 8-PSK,
- možnost volby mezi 1 nebo 2 kanály,
- modul pro konvoluční kódování,
- měření SNR a volba parametrů (např. zesílení) na základě výsledku,
- UART rozhraní – 57,6 kbit/s,
- AES-128 šifrování,
- velikost efektivní hodnoty výstupního proudu až 1 A,
- výstupní napětí pro PLC až 14 V – špička-špička,
- teplotní a proudový senzor,
- pracovní teplota – −40° až +105°.

Služby fyzické vrstvy

- FSK, PSK a jejich módy – modulace i demodulace,
- volba nosné až do 250 kHz,
- určení SNR,
- synchronizace komunikace.

Služby linkové vrstvy

- Zapouzdření uživatelských dat do rámců,
- rozdělení rámců,
- detekce chyb rámců,
- kryptografie pomocí AES 128bit algoritmu,
- monitoring přenosových parametrů.

PSK klíčování Každý typ podporovaných PSK klíčovacích technik umožňuje jinou datovou rychlost. V tab. č. 4 jsou podporované druhy PSK a jejich parametry uvedeny.

Druhem s největší teoretickou bitovou rychlostí na úrovni fyzické vrstvy je modulace 8-PSK. Ovšem při větším zarušení přenosového vedení nemusí být BER (Bit Error Rate) uspokojující. Proto lze použít modulace kódované, kde je vstupní bitová posloupnost kódovaná konvolučním kódováním. V krajním případě je možnost použít u BPSK modulace techniku PNA, která je popsána níže.

Tab. 4: Podporované typy PSK klíčování [12].

PSK klíčování	**Symbolová rychlost [Bd]**	**[bit/symbol]**	**Bitová rychlost [bit/s]**
BPSK	9 600	1	9 600
QPSK	9 600	2	19 200
8-PSK	9 600	3	28 800
Kódovaná BPSK	9 600	0,5	4 800
Kódovaná QPSK	9 600	1	9 600
Kódovaná BPSK + PNA	9 600	0,25	2 400

Technika PNA U BPSK je uvedeno, že lze použít techniku PNA – Peak Noise Avoidance – Technika vyhýbání se impulzním výkyvům na silovém vedení. Jedná se o techniku zajišťující odolnost vůči impulzním výkyvům na silnoproudém vedení. To vyžaduje synchronizaci se signálem na vedení, tzn. příslušný pin ST7580 musí být napojen na obvod, který určuje, kdy se signál na silnoproudém vedení nachází v nule, jinými slovy začíná nová perioda – zkráceně ZC (Zero-Crossing).

FSK klíčování Datovou rychlost v tomto případě určuje tzv. deviační faktor, viz vzorec č. 1. Souhrn podporovaných typů FSK a příslušné deviační faktory viz tab. č. 5. Nutno podotknout, že ve zmíněné tabulce, představuje každý jeden symbol právě jeden bit.

$$\Delta f = \frac{symbolová\_rychlost}{2} \cdot deviační\_faktor \tag{1}$$

Tab. 5: Symbolové rychlosti a deviační faktory pro FSK [12].

Symbolová rychlost [Bd]	**Deviační faktor**	
—	1	0,5
1 200	0,6 kHz	0,3 kHz
2 400	1,2 kHz	0,6 kHz
4 800	2,4 kHz	1,2 kHz
9 600	4,8 kHz	2,4 kHz

Duální kanál Důležitou funkcionalitou čipu ST7580 je podpora přenosu dvěma přenosovými kanály a to jak pro příjem, tak pro vysílání. Nejde o plně duplexní přenos jako takový. Stále se jedná o polo-duplexní variantu přenosu, jelikož lze pracovat v danou chvíli pouze s jedním kanálem. Výhodou použití Duálního kanálu je např. oddělení dvou typů datového provozu. V módu Duálního kanálu naslouchá na obou kanálech a pokud na jednom dekóduje validní rámec, je naslouchání na druhém kanálu ukončeno a modem obstarává příjem na kanále prvním. Obdobná je situace při vysílání, kdy je na jednom kanálu prováděno vysílání, zatímco druhý je neaktivní.

Dále lze v čipu nastavit i možnost, který kanál je výhradně určen pro vysílání a který pro příjem. V tab. č. 6 jsou uvedeny všechny možné kombinace klíčovacích technik v módu Duálního kanálu.

Tab. 6: Povolené kombinace modulací pro jednokanálový a dvoukanálový příjem [12].

Přijímací mód	**Kanál s vyšší frekvencí**	**Kanál s nižší frekvencí**
Jednokanálový	Jakýkoli typ PSK	—
	Vybraný typ FSK	—
Dvoukanálový	Jakýkoli typ PSK	Jakýkoli typ PSK
	Vybraný typ FSK (maximální symbolová rychlost nebude vyšší než 2400 Bd)	Jakýkoli typ PSK
	Jakýkoli typ PSK	Vybraný typ FSK (ale max. symbolová rychlost nebude vyšší než 2400 Bd)

Frekvenční odstup jednotlivých kanálů Pro korektní provoz v módu Duálního kanálu se musí ovšem zachovat následující pravidla, tak aby příjem a vysílání probíhaly správně, kde f_1 je kanál s nižší frekvencí a f_2 je kanál s vyšší frekvencí:

- **Minimální frekvence nosné:** 9 kHz,
- **maximální frekvence nosné pro vysílání:** 250 kHz,
- **maximální centrální frekvence:**

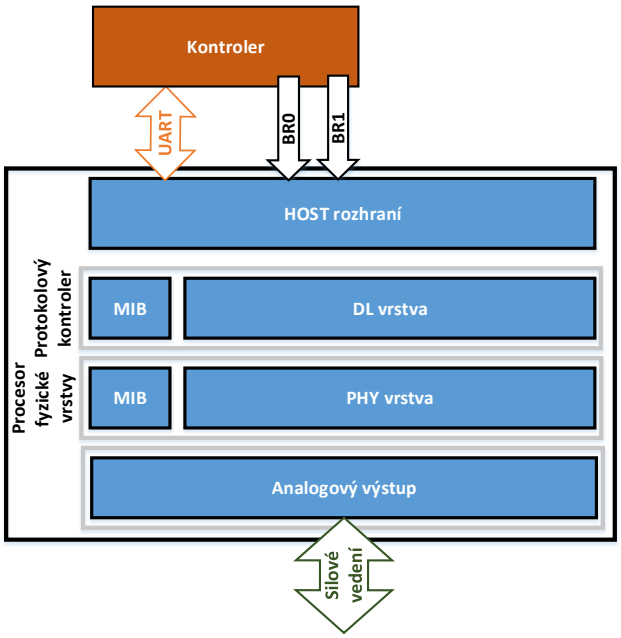
$$\frac{f_1 + f_2}{2} = 249,999\,kHz, \tag{2}$$

- **maximální frekvenční rozdíl:**

$$f_1 - f_2 = 38,461\,kHz. \tag{3}$$

Vnitřní uspořádání ST7580 Zde je toto uspořádání rozebráno pouze stručně, ale v [12] jsou objasněny všechny aspekty ST7580. Na obr. č. 4 je uvedeno vnitřní uspořádání čipu ST7580 spolu s komunikačními rozhraními.

Služby fyzické a linkové vrstvy byly již uvedeny výše. O jejich vykonání se starají "Procesor fyzické vrstvy"a "Protokolový kontrolér". Na zmíněném obrázku jsou dále uvedeny tzv. MIB bloky – tyto bloky reprezentují část paměti, ve které se udržují informace o aktuální konfiguraci systému. Jejich modifikací tedy měníme nastavení systému (např. volba klíčování, AES klíč). Na konci řetězce stojí modem, respektive analogový výstup, který na výstupu poskytuje výsledný analogový datový signál přidávaný na silové vedení.



Obr. 4: Blokové schéma čipu ST7580.

HOST rozhraní S UART, respektive HOST rozhraním na obr. č. 4 souvisí dva piny BR0 a BR1. Na obou z těchto pinů lze nastavit logickou 0 či 1, což dává 4 kombinace nastavení a odtud ST7580 zjistí, jakou rychlostí bude komunikace s externím zařízením na UART probíhat – výpis kombinací nastavení viz tab. č. 7. UART rozhraní je připojené na blok HOST, který je reprezentací modulu zajišťující komunikaci ST7580 s externím zařízením. Jiná možnost kontrolovat zařízení z vnějšího prostředí není.

Tab. 7: Kombinace nastavení pinů BR0 a BR1 [12].

BR0	**BR1**	**[bit/s]**
0	0	9 600
0	1	38 400
1	0	19 200
1	1	57 600

MIB tabulky Tyto tzv. MIB tabulky (Management Information Base) slouží k uchovávání konfigurace čipu ST7580. Jejich nastavením měníme nastavení ST7580 i podobu PLC komunikace. V příloze č. B jsou uvedeny všechny dílčí MIB tabulky a zde je uvedena hlavní MIB tabulka, viz tab. č. 8. Systém obsahuje 10 MIB tabulek, z nichž každá má jiná přístupová práva z pohledu ovládání externím MCU připojeného pomocí UART na HOST rozhraní – "R" jako "Read" (určeno pouze pro čtení), "W" jako "Write" (lze i zapisovat) a "E" jako "Erase" (lze i mazat příslušná data).

Tab. 8: Hlavní MIB tabulka [12].

Index	**Název**	**Velikost [ms]**	**Oprávnění**
0x00	Konfigurace modemu	1	R/W
0x01	Konfigurace PHY	14	R/W
0x02	SS klíč	16	R/W
0x03	Rezervováno	1	R
0x04	Poslední indikovaná data	4	R
0x05	Informace o poslední potvrzovací zprávě	5	R
0x06	PHY Data	10	R/E
0x07	DL Data	8	R/E
0x08	SS Data	10	R/E
0x09	HI časové intervaly	3	R/W
0x0A	Verze firmwaru	4	R

2 ROZBOR ZADÁNÍ A METODIKA ŘEŠENÍ

V této kapitole je nejprve provedena analýza zadání a cílů práce. Po-té je navrženo základní blokové zapojení a členění na jednotlivé funkční celky a jsou ustanoveny příslušná komunikační rozhraní a protokoly. Dále je uveden postup průběhu vypracovávání práce.

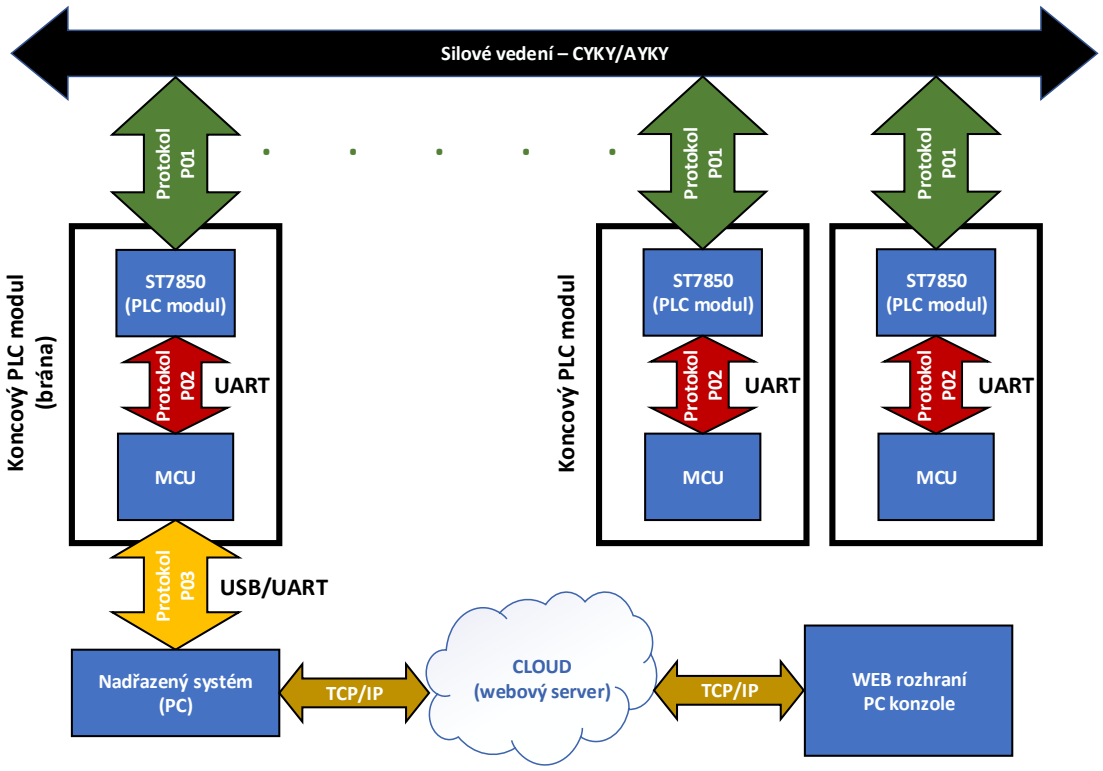
2.1 Analýza zadání

Pro realizaci byly zvoleny výše uvedené HW součásti. Vytvořený systém je určen např. pro nasazení v rodinném domě jako signalizační/monitorovací síť. Nejedná se o datovou síť jako ethernet/WiFi ani o variantu popsanou v práci [7], kde od strany 24 autor rozebírá problematiku návrhu gigabitové PLC sítě, konkurující zmíněným LAN technologiím.

Tato práce řeší problematiku PLC sítě, která je určena pro přenos malého objemu dat. Čímž je komunikace např. mezi senzory či relé, které ovládají např. alarmy/topení. Jedna z možných aplikací sítě je v chytrých domech pro ovládání domovních součástí, jako jsou rolety či topení. Pro řízení prvků postačuje přenášet data rychlostí v řádu stovek B/s.

Řešený systém je monitorován z nadřazeného systému. Dále se nadřazený systém označuje zkratkou NS. Tzn. v síti musí být umístěn modul, který tuto komunikaci zajišťuje. Analogicky je možné demonstrativně síť ovládat z NS.

Hlavní blokové schéma (BS) zapojení navržené na základě analýzy je uvedeno na obr. č. 5. Zde je zobrazeno několik částí, které jsou níže popsány spolu se vztahem mezi nimi.



Obr. 5: Hlavní blokové schéma práce.

2.2 Koncept celkového zapojení

Nyní je objasněn základní princip činnosti vytvořeného konceptu z pohledu jednotlivých modulů a bloků. Jedná se zejména o blok Koncový PLC modul a blok Nadřazený systém.

Koncový PLC modul Hlavními prvky jsou koncové PLC moduly připojené přímo na silové vedení. Moduly mezi sebou komunikují na základě protokolu označeného jako **P01**.

Jsou vždy složeny ze dvou pod-částí – čipu ST7580 (kitu osazeného tímto čipem), který obstarává komunikaci po silnoproudém vedení na úrovni fyzické a částečně i linkové vrstvy. Tento čip zapouzdří data, která přijme od druhé části koncového modulu – MCU. Obě části jsou spolu propojeny pomocí poloduplexní asynchronní sériové linky (UART – Universal Asynchronous Receiver and Transmitter). Protokol má zde pracovní označení **P02**.

Jeden koncový PLC modul obstarává komunikaci s NS. Toto zařízení je označeno na výše zmíněném obrázku jako **brána** a jeho funkcí je odesílat hlášení NS a přijímat příkazy od NS. O to se stará MCU jednotka v příslušném koncovém PLC modulu, která je propojena s PC pomocí USB sběrnice. Protokol je označen jako **P03**.

Nadřazený/externí systém Zprávy obdržené od brány nadřazený systém dále přeposílá na webový server, kam se mohou uživatelé přihlásit pomocí několika platforem a monitorovat tak síť či sledovat data vysílaná zařízeními v síti. Pod tím si lze představit monitoring teploty v bytě či stav bezpečnostních senzorů. Další funkcí může být např. ovládání zařízení jako je topení či klimatizace na dálku.

Komunikace od nadřazeného systému přes webový server až k jednotlivým uživatelským platformám probíhá pomocí standardního TCP/IP modelu. V práci nejsou zpracovány všechny platformy, ale pouze demonstrace principu pomocí jedné platformy – webové stránky. Ta se na základě výše zmíněného obrázku bude připojovat na webový server, kde budou uchovávána data PLC sítě. Ještě je nutné zmínit, že nadřazený systém, který přeposílá zprávy na webový server, je reprezentován konzolovou aplikací na PC.

2.3 Souhrn cílů diplomové práce

Níže jsou uvedeny cíle, jejichž řešením by se měla práce zabývat a které budou v závěrečných kapitolách práce diskutovány. Byly sestaveny následující cíle:

1. Návrh síťové architektury PLC části.
2. Na základě architektury návrh protokolu P01.
3. Popis protokolu P02 a knihovny pro jeho implementaci.
4. Návrh protokolu P03.
5. Implementace komunikačních protokolů P01, P02 a P03 a celkový popis firmwaru tvořící tyto implementace.
6. Návrh a následná implementace připojení k nadřazenému systému.
7. Propojení nadřazeného systému s webovým serverem (webovým rozhraním).
8. Vytvoření testovacích scénářů a demonstrace výsledků.

3 NÁVRH ARCHITEKTURY PLC SÍTĚ

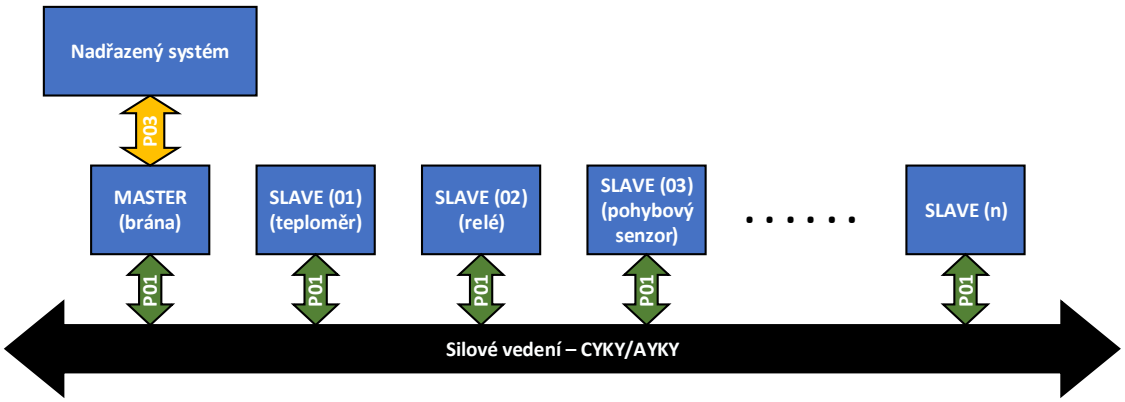
Prvním krokem je návrh architektury PLC sítě. Od něho se odvíjí veškeré návrhy a implementace řešení. Dále je tento síťový model/architektura označovat jako **AS01** (Architektura Sítě číslo 01).

Celkový návrh ilustruje obr. č. 6. Zde lze vidět, že PLC technologie z principu silového vedení pracuje ve sběrnicové topologii. Z toho vyplývá, že zpráva vyslaná jedním účastníkem komunikace je zachycena všemi ostatními účastníky komunikace. Na zmíněném obrázku jsou zobrazeny zařízení připojené na silové vedení, které přes něj komunikují podle pravidel protokolu P01. Tato zařízení jsou, jak je uvedeno na obrázku, dvou typů:

- **MASTER** zařízení,
- **SLAVE** zařízení.

Tyto dva typy zařízení představují Koncové PLC moduly, tak jak jsou uvedeny v kapitole č. 2.2. Z obrázku tedy vyplývá, že se přikročilo ke zvolení centralizované architektury master-slave. Uvažovalo se i např. nad architekturami peer-to-peer či pouze centralizovanou. V knize [13] na straně 31–37 autor přehledně rozebírá jednotlivé aspekty zmíněných architektur pro PLC nasazení. Byla zvolena právě varianta kombinace centralizované architektury s master-slave architekturou.

Nevýhodami jsou např. požadavky na master zařízení, které je díky centralizované architektuře bodem, přes který probíhá veškerá síťová komunikace. Druhou nevýhodou může být, že je nutná velká režijní redundance. Náměty to jsou opodstatněné, ale spíše by se jednalo o problém v síti určené pro přenos velkého objemu dat odpovídající přenosovou rychlostí, či v systémech vyžadující práci v reálném čase. Ani jedna ze zmíněných nevýhod, by ale neměla být problém v navrhované síti, jelikož se jedná o síť vyžadující pouze rychlosti v řádu stovek B/s a navíc se nepředpokládá aplikace vyžadující minimální zpoždění v přenosu dat.



Obr. 6: Komunikační model pro silovém vedení (AS01).

3.1 Popis návrhu architektury AS01

Nejprve je nutné objasnit funkci a účel obou typů zařízení, vyskytujících se v síti AS01. V síti se tedy mohou nacházet pouze dva typy zařízení:

- **MASTER Device (MD)**,
- **SLAVE Device (SD)**.

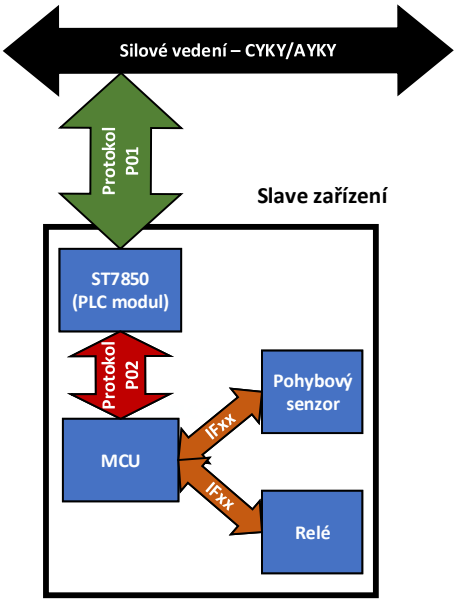
3.1.1 Master zařízení

Toto zařízení je a musí být v síti pouze právě jedenkrát a bude vykonávat především řízení/režii sítě a spojení s nadřazeným systémem:

- přidělovat právo k přístupu na médium tak, aby co nejméně docházelo ke kolizím, a udržovat informaci, kdo má právě právo k přístupu na médium,
- přidělovat adresu nově připojenému SD zařízení na základě volného adresního prostoru, tzn. provádět jeho registraci do sítě,
- udržovat tabulku základních informací o každém připojeném SD zařízení v síti,
- sloužit jako komunikační brána pro nadřazený systém s PLC sítí,
- přijímat příkazy od nadřazeného systému a rozhodovat o jejich vykonání, tzn. MD bude **vždy** upřednostňovat režii PLC sítě, takže nemusí být nadřazenému systému pokaždé odpovězeno okamžitě či odpovězeno vůbec.

3.1.2 Slave zařízení

Zatímco MD obstarává především režii sítě a funkce brány, tak SD naopak ani jednu z těchto funkcí nevykonává a pouze dbá pokynů MD či žádá MD o práva k vysílání. Příklad vnitřního zapojení SD zařízení je uvedeno na obr. č. 7.



Obr. 7: Příklad zapojení SD zařízení.

Na výše uvedeném obrázku je zobrazeno, že jsou na jednotku MCU připojeny další dva přídavné moduly – konkrétně pohybový senzor a relé – pomocí odpovídajícího rozhraní, které MCU podporuje (pro pořádek je rozhraní označeno IFxx a nejedná se o žádný konkrétní typ rozhraní). Tyto připojené přídavné moduly stanovují bližší určení SD a do určité míry i konkrétní firmware příslušného SD MCU. Slave zařízení se dále dělí podle typu připojených přídavných modulů z pohledu MD do dvou skupin:

- **Odpovídající SD zařízení** – zařízení pouze odpovídá na dotazy od MD; zde lze zařadit SD zařízení s přídavným teploměrovým modulem.
- **Vykonávající SD zařízení** – zařízení pouze provádí požadovaný příkaz od MD; příkladem může být přídavný relé modul ovládající topení.

Na základě skupiny a konkrétního typu připojeného přídavného modulu je uzpůsobeno složení dat v rámci sítě AS01. Pro každý typ SD musí být v MD zařízení definován způsob komunikace s příslušným typem SD zařízení. Ovšem veškeré ostatní síťové požadavky zůstávají stejné a v konečném důsledku půjde tedy jen o skladbu datového pole ve výsledném AS01 rámci.

3.2 Přenosové parametry sítě AS01

Komunikace bude probíhat po dvou kanálech (viz mód duálního kanálu v podkapitole č. 1.2.2). Každý bude modulován jiným způsobem a bude mít jiné určení. Půjde o kanál s nižší frekvencí (datový kanál), který je označován jako LFC (Low Frequency Channel) a o kanál s vyšší frekvencí (režijní kanál) s označením HFC (High Frequency Channel).

Spojení bude poloduplexní, jelikož na základě vlastností ST7580, PLC modul může v danou chvíli pouze vysílat nebo přijímat. K rozdělení na LFC a HFC se přistoupilo za účelem oddělení uživatelských dat od režie sítě. Pro režii sítě není potřeba vysoká datová rychlost, takže se může přistoupit k použití robustní modulace, aby režie sítě mohla probíhat i při větším rušení. Popis jednotlivých kanálů:

- **Datový kanál LFC:**
  - přenos uživatelských dat,
  - větší bitová rychlost než HFC.
- **Režijní kanál HFC:**
  - přenos režijních síťových dat,
  - robustní modulace,
  - malý datový tok,
  - pouze režijní data.

Na PLC modulu lze nastavit příslušné fyzické a linkové parametry komunikace na základě MIB (konkrétní tab. viz MIB tab. č. 43, 46, 47 a 49). Toto nastavení musí být aplikováno v každém koncovém PLC modulu, aby byl zajištěn korektní přenos dat. Tyto parametry jsou identické jak pro MD, tak pro SD zařízení.

V tab. č. 9 jsou parametry přenosu dat z pohledu využitých klíčovacích technik a zvolených frekvencí. Nastavení týkající se linkové vrstvy PLC modemu jsou uvedeny v tab. č. 10. V obou tabulkách jsou uvedeny pouze parametry, u kterých buď byla provedena změna oproti továrnímu nastavení, nebo které jsou klíčové ke správnému fungování sítě. Údaje z těchto dvou tabulek jsou zapsány do MIB tabulky Konfigurace modemu a Konfigurace fyzické vrstvy PHY.

Aby PLC modul neposílal ve svých Status zprávách, které jsou zmíněny v pod-části kapitoly č. 4.2.4, příznak značící špatnou rekonfiguraci modulu je nutné současně se zmíněnými dvěma MIB tabulkami změnit i MIB s názvem SS klíč (viz tab. č. 49). Tato MIB, obsahuje 128bit klíč používaný k šifrování při použití šifrovaného přenosu. Komunikace v AS01 prozatím šifrovaná nebude, ale pro správné fungování vyžaduje PLC modul změnu z továrního nastavení (které je 16 bytů po sobě jsoucích, každý s hodnotou 0x00) na jakoukoli jinou hodnotu – zde byla zvolena posloupnost střídání nul a jedniček, což odpovídá 16 bytů, kde každý nabývá hodnoty 0xAA.

Tab. 9: Nastavení fyzických přenosových vlastností sítě AS01.

Parametr	**Kanál s nižší frekvencí (LFC)**	**Kanál s vyšší frekvencí (HFC)**
Frekvence	75 kHz	95 kHz
Klíčování	8-PSK	BPSK
Symbolová rychlost	9 600 Bd	9 600 Bd
Bitová rychlost	28 800 bit/s	9 600 bit/s

Tab. 10: Nastavení linkové vrstvy sítě AS01.

Parametr	**Nastavení**
Délka CRC	4 B
Sniffer mód	neaktivní
Vrstva, jejichž data PLC modul exportuje k MCU	DL vrstva
Rozsah dat pro výpočet CRC	DL Data + PHY Length
Délka PRE pole v PSK rámci	4 B

3.2.1 Časové intervaly

Je nutné uvést několik časových intervalů používaných v této síti. Jsou uvedeny v tab. č. 11, kde je uveden i jejich popis a hodnota. Jedná se o časové intervaly:

- **TTR** – z anglického Time Token-Resend,
- **TLT** – z anglického Time Lease-Token,
- **THPE** – z anglického Time High-Priority-Event.

Tab. 11: Časové intervaly používané v síti AS01.

Označení	**Hodnota [ms]**	**Popis**
TTR	1000–2000	Časový interval pro opakované zaslání žádosti o token a pro opakování žádosti registrace do sítě AS01
TLT	400	Doba platnosti tokenu
THPE	100	Doba pro pakované vysílání zprávy typu `HIGH_PRIORITY_EVENT`, při jejím nepotvrzení

3.3 Přístup k médiu

Řešení přístupu řídí jen a pouze MD pomocí tzv. tokenu – kdo má token, tak vysílá, přičemž token je pouze jeden. Pokud chce SD zahájit komunikaci, použije LFC k žádosti o vysílání a je jen na MD, zda žádosti vyhoví. Jako neakceptování žádosti je považována i situace, že žádající SD od MD odpověď vůbec nedostane; tzn. zpráva neakceptování žádosti se neposílá.

Pokud SD nedostane odpověď, dál provádí svou funkci a po určitém časovém intervalu TTR se pokusí vyžádat od MD token opakovaně. Následně se situace opakuje, dokud SD nedostane přidělen token a neodešle data. Pokud SD token nevrátí MD v určitém časovém intervalu TLT, token automaticky vyprší a MD považuje token za navrácený.

Na každý token může SD odeslat pouze **jednu** zprávu. To je vyhovující, jelikož se nepředpokládá přenos velkého množství dat. Tímto krokem se též ulehčí režii sítě, jelikož MD zařízení bude po odeslání právě jedné zprávy SD zařízením předpokládat, že token s tuto zprávou SD vrací MD zařízení.

3.4 Potvrzování komunikace

SD zařízení potvrzuje veškerou komunikaci přicházející od MD a MD potvrzuje příjem jen v některých případech. Lépe je problematika potvrzování vysvětlena v kapitole č. 4.1.3, jelikož od typu posílané zprávy se odvíjí, zda má být potvrzena či nikoli. Obecným principem pro potvrzování zpráv je použita níže popsaná technika ACK-After-All (dále AAA).

ACK-After-All Technika potvrzování AAA není oficiální technika potvrzování, technika byla takto pojmenována a navržena pouze za účelem této práce. Principem techniky je potvrzovat komunikaci, až je to bezprostředně nutné.

Např.: MD odešle k příslušnému SD žádost o zaslání dat. Tuto žádost SD nepotvrzuje, ale odešle příslušná data, a ta se berou jako potvrzení. Pokud je zpráva žádosti ve SD zařízení chybně dekódovaná, odešle SD potvrzení, které značí chybnou zprávu, a MD odešle žádost znova. Pokud se odpověď s daty od SD dostane k MD a je správně dekódovaná, tak MD již potvrzení nezasílá. Tzn. MD komunikaci nepotvrzuje pokaždé, ale SD vždy. Pokud je zasláno potvrzení chybového přenosu, MD opakuje vysílání zprávy.

3.5 Adresace a přidělování adres

Adresa každého SD a MD zařízení bude mít velikost 1 B. To dává adresní prostor 256 adres. Adresy budou přidělovány MD zařízením, přičemž MD bude mít svou adresu danou továrním nastavením a bude **neměnná**. V tabulce č. 12 jsou uvedeny rozsahy adres pro jednotlivé aplikace.

Při připojení nového SD do sítě, SD zařízení pravidelně vysílá žádost o připojení do sítě na továrně definovanou adresu MD zařízení. Dokud SD nedostane přidělenou adresu od MD zařízení, vystupuje pod předem definovanou adresou, která je určená pro nově připojované SD zařízení (v SD továrně zapsána). Takto MD pozná, že SD zařízení nejeví chybu, když vysílá bez tokenu, ale že se jedná o zařízení žádající připojení k síti.

Tab. 12: Adresace sítě AS01.

Účel adresy	**Adresa / adresní rozsah**	**Kódové označení**
Master zařízení	0x00	`MASTER_ADDRESS`
Slave zařízení	0x10 až 0xDE (195 adres)	—
Broadcast vysílání	0xFF	`BROADCAST`
Zvláštní využití	0xF0 až 0xFE (15 adres)	—
Výchozí adresa neregistrovaného SD	0xFA	`NON_SUBS_ADDRESS`

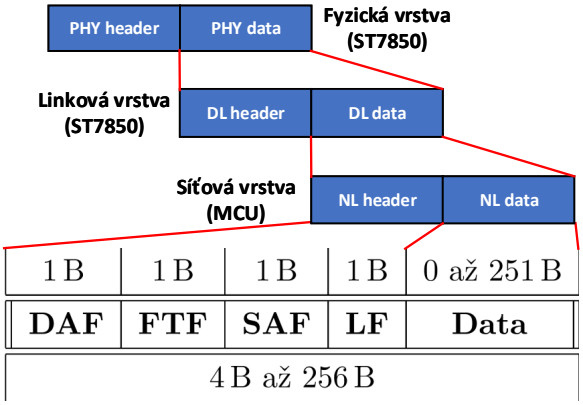
4 NÁVRH KOMUNIKAČNÍCH PROTOKOLŮ

Tato kapitola se zaměřuje na návrh všech tří zmíněných komunikačních protokolů. Respektive jde o částečný návrh P01 (proč se jedná o částečný je uvedeno v úvodem následující podkapitoly č. 4.1). Následuje pouze popis P02, jelikož je již definován výrobcem čipu. Nakonec je proveden kompletní návrh posledního protokolu v řetězci P03. Na úvod je nutné uvést tři důležité zkratky, které jsou dále v následující práci hojně využívány. Shrnutí důležitých zkratek pro další text:

- **PHY** (Physical) – fyzická vrstva komunikačního modelu,
- **DL** (Data Link) – linková vrstva komunikačního modelu,
- **NL** (Net Layer) – síťová vrstva komunikačního modelu.

4.1 Komunikační protokol PLC sítě

Prvním protokolem je protokol přenášený přes PLC síť a je označován jako **P01**. Jedná o částečný návrh, což je ilustrováno na obr. č. 4.1. Přímo na vedení je ST7580 připojen pomocí vazebního členu a vysílá rámce fyzické vrstvy – jak již bylo zmíněno, ST7580 obstarává služby fyzické a linkové vrstvy a MCU pouze dodává data, která budou čipem ST7580 zapouzdřena jako data DL rámce. Samotný přínos práce tedy zde spočívá v návrhu rámce NL vrstvy, který představuje ona zapouzdřovaná data.



Obr. 8: Zapouzdření rámců sítě AS01.

4.1.1 Fyzická vrstva

Pro popis PHY rámce a obecně fyzické vrstvy implementované v ST7580 se vychází z oficiálního manuálu na webových stránkách výrobce čipu, dostupného z [9]. Na úvod je nutné uvést, že rámec fyzické vrstvy je pro každou klíčovací techniku odlišný. Jedná se dva druhy rámců pro:

- **A) PSK klíčování**,
- **B) FSK klíčování**.

A) PSK klíčování

Rámec pro PSK klíčování je popsán níže. V tab. č. 4 v předchozí části práce jsou uvedeny všechny módy PSK modulace, které čip ST7580 podporuje.

Skladba rámce Na obr. č. 4.1.1 je uvedena podoba PSK rámce na úrovní fyzické vrstvy. Při použití PSK je pole PRE, UW a Mód modulováno na nosný signál BPSK modulací a zbytek je modulován PSK technikou, jejíž kód je uveden v poli Mód. Příjemce tedy musí nejprve demodulovat první část, aby následně mohl nastavit správnou techniku pro demodulaci. Význam jednotlivých polí PSK rámce a jejich zkratky:

- **Preambule** (PRE) – pole slouží jako synchronizace a obsahuje sekvenci nul a jedniček, které se vzájemně střídají (ekvivalentní ke znaku 0xAA); slouží k synchronizaci.
- **Unikátní znak** (UW) – Unique Word; je předdefinováno; určuje začátek samotného rámce; používá se dále volitelně k určení SNR.
- **Mód** – znak, který určuje, jaká modulace byla použita pro modulaci SDU.
- **SDU** – data fyzické vrstvy; délka je vždy alespoň 1 B, jelikož první byte obsahuje informaci o délce SDU.

2 B až 5 B	4 B	1 B	1 B až 255 B
Preambule (PRE)	**Unikátní znak (UW)**	**Mód**	**SDU fyzické vrstvy**
BPSK modulace			Modulace určená v poli Mód

Obr. 9: Podoba PSK rámce dle P01 [9].

B) FSK klíčování

Možnosti symbolových rychlostí a deviačních faktorů byly uvedeny v tab. č. 5.

Skladba rámce Na obr. č. 4.1.1 je uvedena skladba FSK rámce. Na rozdíl od PSK je celý rámec modulován stejnou technikou. Tyn. vysílači i přijímači musí být předem znám použitý typ FSK, UW, deviační faktor a symbolová rychlost. Význam polí je:

- **Preambule** – totožný účel a činnost jako u PRE v PSK rámci.
- **Unikátní znak** – volitelná hodnota značící začátek SDU; volba délky 1 B nebo 2 B.
- **SDU** – data fyzické vrstvy; význam stejný, jako u PSK rámce.

2 B až 5 B	1 B až 2 B	1 B až 256 B
Preambule (PRE)	**Unikátní znak (UW)**	**SDU fyzické vrstvy**
FSK modulace		

Obr. 10: Podoba FSK rámce dle P01 [9].

4.1.2 Linková vrstva

Čip ST7580 nabízí i některé služby DL vrstvy pro PLC komunikaci a ty jsou uvedeny v podkapitole č. 1.2.2. Opět při popisu DL rámce a DL služeb se vychází z uživatelského manuálu vydávaného výrobcem čipu [9]. V této aplikaci je využíváno následujících služeb:

- zapouzdření dat určených k vyslání na silové vedení do odpovídajících rámců,
- korektní rozdělení většího objemu dat do rámců,
- detekce chybně přijatých rámců,
- zaznamenávání parametrů datového přenosu.

Skladba rámce DL vrstvy Skladba rámce viz obr. č. 4.1.2. Význam jednotlivých polí:

- **Length** – velikost pole Data v bytech.
- **Data** – data určená k přenosu.
- **CRC** – výsledná hodnota CRC kontrolního součtu (která data se zahrnou do kontrolního součtu je volitelné, viz MIB).

1 B	0 B až 255 B	1 B nebo 2 B nebo 4 B
Length	**Data**	**CRC**

Obr. 11: Rámec linkové vrstvy [9].

4.1.3 Síťová vrstva

V této podkapitole je popsána realizace vlastního návrhu protokolu, který nese uživatelská či režijní data sítě AS01. Na obr. č. 4.1.3 je zobrazen vytvořený koncept rámce síťové vrstvy a dále je v tab. č. 13 uveden význam a příslušné zkratky jednotlivých polí.

1 B	1 B	1 B	1 B	0 až 251 B
DAF	**FTF**	**SAF**	**LF**	**Data**
4 B až 256 B				

Obr. 12: Rámec síťové vrstvy sítě AS01.

Tab. 13: Popis jednotlivých polí NL rámce P01.

Označení	Význam zkratky	Popis
DAF	Destination Address Field	Adresa zařízení, pro které je rámec určen
FTF	Frame Type Field	Pole nesoucí informaci o účelu rámce
SAF	Source Address Field	Adresa odesílatele rámce
LF	Length Field	Celková délka rámce v bytech
Data	—	Uživatelská či režijní data

Popis skladby NL rámce Rámec síťové vrstvy obsahuje nejprve pole s adresou zařízení, pro které je rámec určen. Toto pole je zařazeno jako první z toho důvodu, že všechna připojená zařízení vyslanou zprávu přijmou a následně jak MCU provádí dekódování přijatých dat, rozhodne již na základě prvního bytu, zda je pro něj zpráva určena a zda má pokračovat v dekódování. Následuje pole FTF, které určuje obsah a typ přenášených dat. Dalším bytem v pořadí je adresa zařízení, odkud byl rámec vyslán, tak aby přijímacímu zařízení bylo jasné, kam má eventuálně zaslat odpověď. Posledním povinným polem pro všechny typy rámců je pole LF, které uvádí informaci o délce celého rámce včetně povinných polí. Na základě LF, MCU zjistí, kdy přestat data dekódovat, pokud je PLC modulem vyexportována zpráva delší než byla odeslána či zpráva kratší (např. rušení na kanále). Tzn. přijatý rámec je zahozen (v případě zprávy kratší) nebo je přijato pouze určité množství dat na základě pole LF (v případě zprávy delší).

Možný obsah polí DAF a SAF je popsán v kapitole č. 3.5. Pole LF je generováno vysílacím MCU na základě délky rámce. Nakonec tedy zbývá tedy popsat možné hodnoty obsažené v poli FTF, na jehož základě je i vytvořena skladba pole Data, viz níže.

4.1.4 Typy zpráv síťové vrstvy

NL rámec obsahuje pole FTF. Toto pole nese informaci o tom, jak se má s danými daty naložit. Tzn. pole FTF definuje skladbu pole Data a obecně typ rámce. Podle FTF lze rozdělit rámce do 4 větších skupin, respektive následně budou označovány jako typy zpráv:

- **A) Datové zprávy**,
- **B) Konfigurační zprávy**,
- **C) Režijní zprávy**,
- **D) Zprávy pro speciální užití**.

A) Datové zprávy

Pomocí datových rámců, se vyměňují data mezi MD a příslušným SD zařízením. Pro každý úkon je definován jiný typ rámce. V tab. č. 14 jsou všechny typy datových rámců uvedeny.

Tab. 14: Typy datových zpráv sítě AS01.

Pracovní označení	Hodnota FTF	Odesílatel	Možná odpověď
DATA_SEND	0x10	SD	DATA_ACK DATA_NACK
DATA_REQUEST	0x11	SD	DATA_WRITE DATA_NACK
DATA_READ	0x12	MD	DATA_SEND DATA_NACK
DATA_WRITE	0x13	MD	DATA_ACK DATA_NACK

Pro datovou komunikaci byl vyčleněn kanál s nižší frekvencí, takže všechny datové rámce jsou vysílány skrze LFC. Skladba pole Data se liší v závislosti na typu odesílaných dat, respektive typu SD zařízení (určuje ho typ přídavného modulu). Proto nelze definovat obecné znění pole Data pro datové rámce. V tab. č. 14 jsou definovány odpovědi jako `DATA_ACK` a `DATA_NACK` a jejich význam je rozebrán v podkapitole č. 4.1.4. Na základě zmíněné tabulky datových zpráv jsou k dispozici 4 varianty datových zpráv:

1. **`DATA_SEND`** – jedná se o zprávu obsahující data, která si MD vyžádal od určitého SD, tzn. je to odpověď na zprávu `DATA_READ`.
2. **`DATA_REQUEST`** – zpráva, kterou SD žádá o určitá data MD zařízení; MD následně odpovídá zprávou `DATA_WRITE`, která obsahuje příslušná data.
3. **`DATA_READ`** – žádost vysílající MD k určitému SD, za účelem vyčtení jeho specifických dat; SD odpovídá zprávou `DATA_SEND`.
4. **`DATA_WRITE`** – vysílá MD k určitému SD, jako zprávu obsahující data k zápisu do zařízení; nejedná se o konfigurační data, ale např. o signál k zahájení určité operace (např. sepnutí relé).

B) Konfigurační zprávy

Konfigurační zprávy slouží k výčtu konfigurace přídavného modulu v SD zařízení, nebo k jeho rekonfiguraci. Skladba pole Data je pro tyto zprávy pevně dána na 8 B. Konfigurační zprávy jsou posílány skrze LFC. Rozlišují se tři typy konfiguračních zpráv, které jsou uvedeny v tab. č. 15. Význam jednotlivých typů zpráv:

1. **`SLAVE_CONFIG_READ`** – vysílána MD za účelem vyčtení dat ze SD; odpovědí je buď potvrzení chybného příjmu `DATA_NACK` nebo příslušná konfigurační data ve zprávě **`SLAVE_CONFIG_DATA`**; pole Data je zde prázdné (vyčítají se vždy všechna data).
2. **`SLAVE_CONFIG_WRITE`** – příkaz k rekonfiguraci SD zařízení; odpovědí je, že buď rekonfigurace proběhla správně (`DATA_ACK`) nebo neproběhla správně (`DATA_NACK`).
3. **`SLAVE_CONFIG_DATA`** – zpráva obsahující celou aktuální konfiguraci příslušného SD zařízení.

Tab. 15: Konfiguračních zprávy sítě AS01.

Pracovní označení	Hodnota	Odesílatel	Možná odpověď
`SLAVE_CONFIG_READ`	0x21	MD	`SLAVE_CONFIG_DATA` `DATA_NACK`
`SLAVE_CONFIG_WRITE`	0x22	MD	`DATA_ACK` `DATA_NACK`
`SLAVE_CONFIG_DATA`	0x23	SD	—

C) Režijní zprávy

Režijní zprávy jsou skupinou zpráv, které zajišťují fungování sítě jako takové. Patří sem potvrzovací zprávy, zprávy řízení přístupu na médium a zprávy přihlašování nově připojovaného SD zařízení. Až na zprávy `DATA_ACK` a `DATA_NACK`, které jsou určené pro potvrzování na kanálu pro datové účely, jsou všechny ostatní zmíněné zprávy přenášeny přes režijní kanál.

Všechny typy režijních zpráv jsou uvedeny v tab. č. 16 a nyní je uveden jejich popis a možné odpovědi:

1. `DATA_ACK` / `DATA_NACK` – jako jediné z této skupiny jsou přenášeny po datovém kanálu; zprávy potvrzování datové komunikace; `DATA_ACK` značí bezchybový příjem rámce a `DATA_NACK` chybný příjem rámce, ale je odesíláno pouze pokud byla korektně rozpoznána zdrojová adresa zprávy a typ zprávy; pole Data má délku 1 B a je jím kód potvrzovaného rámce.
2. `MANAGEMENT_ACK` / `MANAGEMENT_NACK` – potvrzovací zprávy pro režijní kanál; pole data a princip odeslání NACK stejný jako u datové potvrzovací zprávy.
3. `TOKEN_REQUEST` – zpráva odesílaná SD zařízením jako žádost o přidělení tokenu od MD, aby mohl začít vysílat na médium; jedinou možnou odpovědí je zpráva typu `TOKEN_TOKEN`, která značí příjem tokenu a nyní SD zařízení může odeslat přesně jednu zprávu; pokud není na tuto zprávu odpovězeno, opakuje se žádost za náhodný čas, který je v rozpětí určitého časového intervalu s označením TTR; po odeslání právě jedné zprávy je MD zařízením předpokládáno, že je token vrácen.
4. `TOKEN_RETURN` – zpráva dobrovolného navrácení tokenu.
5. `TOKEN_TOKEN` – zpráva dávající právo vysílat aneb zpráva obsahující token; odpovědí je potvrzovací zpráva pro režijní kanál.
6. `TOKEN_FORCE_PULL` – MD násilně odebere token příslušnému SD zařízení.
7. `SUBS_REQ` – zpráva pravidelně odesílaná při inicializaci nově připojeného SD na adresu MD; je to žádost o přidělení adresy a přístup k síti; jedinou možnou odpovědí je zpráva `SUBS_ACCEPTED`; pole Data zde obsahuje 1B informaci, o jaký typ SD zařízení se jedná; dokud SD nemá přidělenou adresu, vystupuje pod továrně nastavenou adresou 0xFA, více viz podkapitola č. 3.5.
8. `SUBS_ACCEPTED` – odesíláno MD jako odpověď na předchozí zmíněnou zprávu; pole Data obsahuje 1B hodnotu přidělenou jako novou adresu pro SD.
9. `SUBS_RESET` – zasílána MD zařízením v případě, že se v síti objeví zařízení s adresou, kterou MD nemá obsaženou ve své tabulce zařízení; další použití je v případě, že dojde ke kolizi adres, tzn. MD určí zařízení, které musí provést proces přihlášení k síti znova; pole Data je prázdné; jedinou možnou odpovědí je zpráva značící zahájení nového procesu přihlašování `SUBS_REQ`.

Tab. 16: Režijní zprávy sítě AS01.

Pracovní označení	Hodnota	Odesílatel	Možná odpověď
DATA_ACK	0xAA	SD	—
DATA_NACK	0xAB	SD	—
MANAGEMENT_ACK	0xBA	MD/SD	—
MANAGEMENT_NACK	0xBB	MD/SD	—
TOKEN_REQUEST	0xCA	SD	TOKEN_TOKEN
TOKEN_RETURN	0xCB	SD	MANAGEMENT_ACK MANAGEMENT_NACK
TOKEN_TOKEN	0xCC	MD	MANAGEMENT_ACK MANAGEMENT_NACK
TOKEN_FORCE_PULL	0xCF	MD	TOKEN_RETURN
SUBS_REQ	0x77	SD	SUBS_ACCEPTED
SUBS_ACCEPTED	0x7A	MD	MANAGEMENT_ACK MANAGEMENT_NACK
SUBS_RESET	0x7F	MD	SUBS_REQ

D) Zprávy pro speciální užití

Tato kategorie zpráv zahrnuje 3 typy zpráv a jsou uvedeny v tab. č. 17. Zpráva typu HIGH_PRIORITY_EVENT, kterou odesílá MD nebo SD, pokud dojde k události, která si žádá neprodlené řešení a je vysílána po režijním kanálu (robustní klíčovací technika). Příkladem může být, že bezpečnostní senzor pohybu zachytí pohyb v místě, kde nemá být a tak chce vyhlásit poplach – odešle HIGH_PRIORITY_EVENT zprávu MD a to spustí alarm (též pomocí zmiňované zprávy) a kontaktuje pomocí GSM majitele. Tato zpráva je odesílána v časových intervalech THPE, dokud není potvrzena režijní potvrzovací zprávou značící, že příslušné zařízení provedlo kroky k reakci na náhlou mimořádnou událost. Pole Data je zde naplněno daty, které odpovídají typu SD zařízení.

Druhým typem zpráv jsou zprávy Ping, pomocí kterých MD testuje, zda je SD zařízení stále připojeno a zda ho nemá vymazat ze své databáze. Odpovědí na PING_REQUEST zprávu je jedinou možností zpráva PING_RESPONSE. Pole Data je v obou případech prázdné.

Tab. 17: Zprávy pro speciální užití sítě AS01.

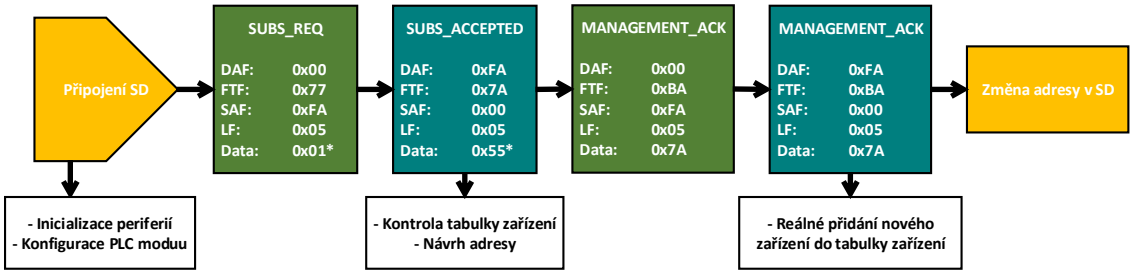
Pracovní označení	Hodnota	Odesílatel	Možná odpověď
HIGH_PRIORITY_EVENT	0xFF	MD/SD	MANAGEMENT_ACK
PING_REQUEST	0x01	MD	PING_RESPONSE
PING_RESPONSE	0x02	SD	—

4.1.5 Příklady komunikace

V této podkapitole jsou uvedeny dva příklady komunikace pomocí protokolu P01.

Připojení nového SD zařízení do sítě AS01

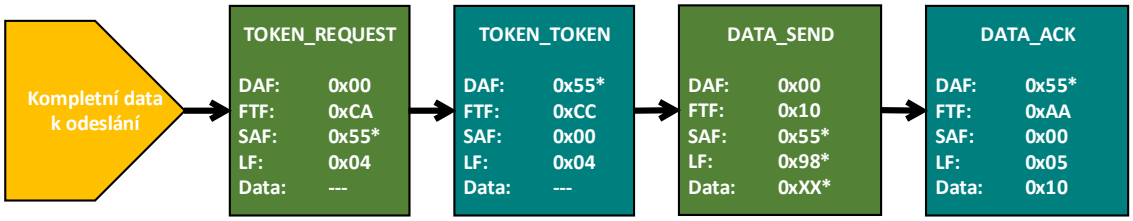
Celý proces je uveden na obr. č. 13[1]. Po připojení SD k síti proběhne inicializace zařízení a spustí se cyklus vysílání žádosti o registraci do sítě. SD zasílá zprávy `SUBS_REQ` v pravidelných časových intervalech TTR, kde pole Data obsahuje kód typu zařízení. Jakmile dostane od MD odpověď zprávou `SUBS_ACCEPTED`, která v poli Data obsahuje návrh adresy, odpovídá režijním ACK, ale stále pod adresou 0xFA určenou pro neregistrovaná zařízení. Jakmile dostane od MD zprávu režijního ACK, předpokládá se, že MD si SD zařízení s konečnou platností zapsalo do své tabulky a SD si svou adresu mění na navrhovanou adresu. Pokud z nějakého důvodu dojde k chybě a SD zařízení si adresu změní, zatímco MD si ji do své tabulky nezapíše nebo naopak – MD zjistí nesrovnalost ihned, jak se rozhodne kontaktovat příslušné SD nebo naopak, a vysílá zprávu `SUBS_RESET`, která resetuje adresu příslušného SD a proces registrace do sítě probíhá od začátku.



Obr. 13: Demonstrace připojení nového SD zařízení do AS01.

Žádost SD zařízení o token a vyslání dat

Na obr. č. 14[1] je proces žádosti SD o token a navazující odeslání dat. SD žádá zprávou `TOKEN_REQUEST` a čeká na odpověď `TOKEN_TOKEN`. Pokud v intervalu TTR token nedostane, žádost opakuje. Jakmile přijme zprávu `TOKEN_TOKEN`, odešle právě 1 zprávu a tímto se považuje token za vrácený. MD následně odpovídá potvrzením korektního/chybného příjmu. Pokud je přijato `DATA_NACK`, celý proces odesílání a žádosti o token se opakuje.

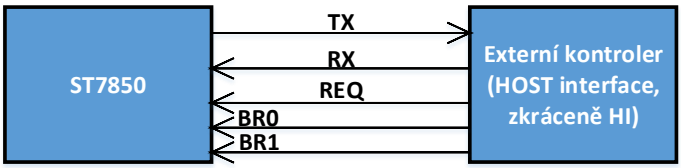


Obr. 14: Demonstrace žádosti o token v síti AS01.

4.2 Komunikační protokol ST7580-MCU

Komunikace mezi MCU a PLC modulem je specifikovaná firmwarem na čipu ST7580 a celý protokol je popsán v uživatelském manuálu [9], ze kterého se v této podkapitole vychází. Tento protokol je označen jako **P02** a musí být implementován přesně, jak je popsán manuálu (jinak není prakticky žádná komunikace možná).

Na úvod je potřeba popsat, jak komunikace s externím zařízením vypadá, co se týče propojení. To lze vidět na obr. č. 15. Komunikace probíhá pomocí UART rozhraní – TX pro vysílání a RX pro příjem z pohledu ST7580. Dále je nutný spoj s označením REQ, pomocí kterého externí MCU vyžaduje komunikaci – ST7580 je vždy master[2] Je vždy jen na master jestli požadavku vyhoví. Jinak se rozhraní pro externí ovládání ST7580 také nazývá HOST rozhraní, zkráceně HI (HOST interface).



Obr. 15: Fyzické propojení mezi MCU a ST7580.

Požadavky na UART linku a následnou implementaci

- Rozlišení mezi několika předdefinovanými typy rámců,
- výše popsaný mechanismus přenosu dat, tzn. správné používání pinu REQ,
- potvrzovací mechanismus pro přijaté rámce,
- v komunikaci se musí dodržovat několik časových intervalů,
- ověřování celistvosti rámců a syntaktické korektnosti v každém poli rámce,
- parametry komunikace (např. bitových rychlosti či složení symbolu), tak jak jsou uvedeny ve zmíněném manuálu.

4.2.1 Časové intervaly

Protokol P02 pracuje s časovými intervaly, které synchronizují komunikaci – viz tab. č. 18.

Tab. 18: Synchronizační časové intervaly [9].

Název	**Výchozí hodnota [ms]**
TIC	10
TACK	40
TSR	200

[1]Hodnoty označené symbolem "∗" jsou dány náhodně z podstaty typu daného pole.
[2]POZOR ! Neplést s master zařízením v síti AS01 !

Význam jednotlivých časových intervalů:

1. **TIC** (Time Inter-Character) – časový interval začíná po dokončení vysílání jednoho znaku, během kterého musí být zahájeno vysílání znaku následujícího.
2. **TACK** (Time Acknowledgement) – časový interval začíná po dokončení přijetí Lokálního rámce jedním účastníkem, kdy druhý účastník komunikace čeká na přijetí potvrzovacího rámce, zatímco odeslal Lokální rámec.
3. **TSR** (Time Status-message-Response) – časový interval začíná po přijetí zprávy Status, během kterého musí být zahájen příjem Lokálního rámce master zařízením.

4.2.2 Synchronizace komunikace

UART je asynchronní, proto jsou použity mechanismy, aby nedocházelo ke kolizím a komunikace se synchronizovala. Jedná se o dvě pravidla (platí pro obě strany komunikace).

Pokud jsou obě pravidla zachována a rámec je přijat, je spočítán kontrolní součet a zkontrolována délka Datového pole (hodnota v poli Length). Jestliže jsou obě kontroly správné, je rámec brán jako korektně přijatý. Těmito dvěma pravidly jsou:

1. **Délka rámců** – pokud počet přijatých bytů přesáhne očekávanou hranici na základě formátu rámců, je příjem ukončen,
2. **Časové intervaly** – časový interval mezi jednotlivými znaky, který je označován jako TIC (po uplynutí tohoto intervalu už není žádný charakter akceptován).

Složení znaku Jeden symbol je složen z **8** datových bitů. Znak se vytvoří přidáním Start bitu na začátek symbolu a Stop bitu na konec symbolu. Demonstrace viz obr. č. 4.2.2. Tyto znaky pak tvoří jednotlivé pole v rámcích.

Start bit	**D0**	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**Stop bit**
	8 datových bitů = symbol								
1 znak									

Obr. 16: Složení znaku [9].

4.2.3 Lokální rámec

Jedná se o obecný tvar rámce přenášeného mezi ST7580 a MCU. Jeho podoba je uvedena na obr. č. 4.2.3. Význam jednotlivých polí pak viz tab. č. 19. Jinak tento rámec může být označován zkratkou LOC (Local Frame).

STX	**Length**	**Příkazový kód (CC)**	**Data**	**Kontrolní součet**

Obr. 17: Lokální rámec P02 [9].

Tab. 19: Význam polí Lokálního rámce [9].

Pole	Délka [B]	Hodnota	Popis
STX	1	0x02 nebo 0x03	Začátek rámce
Length	1	0 až 255	Délka pole Data v bytech
Kód příkazu	1	0 až 0xFF	Konstantní hodnota daná pro každý typ vyžadované operace
Data	0 až 255	—	Přenášená data
Kontrolní součet	2	—	Součet pole Length až po poslední byte v Data poli; pokud je Data pole prázdné, zahrneme pole Kód příkazu (v Data poli jdeme byte po byte)

4.2.4 Speciální typy lokálních rámců

Potvrzovací rámce a Status zprávy jsou zvláštním případem Lokálního rámce. Mají své specifické úlohy, které jsou důležité pro korektní průběh komunikace s ST7580.

Potvrzovací rámec Jakmile MCU nebo ST7580 přijme Lokální rámec, musí potvrdit přijetí pomocí Potvrzovacího rámce. Formát rámce je pro obě strany stejný a je dlouhý 1 B. V případě korektního příjmu se pošle potvrzení o bezchybném příjmu (ACK; Acknowledgement), v případě chybného příjmu potvrzení o chybném příjmu (NACK; Non-Acknowledgement). Oba typy jsou uvedeny v tab. č. 20 i s jejich danými hodnotami.

Tab. 20: Význam polí Potvrzovacího rámce P02.

Typ potvrzení	Význam	Kód
ACK	Bezchybový příjem Lokálního rámce	0x06
NACK	Alespoň jedna chyba v příjmu Lokálního rámce	0x15

Status zpráva Je vyslána k externímu MCU vždy, kdy je na pinu REQ změněno napětí na logickou 0. Je složena ze dvou bytů. První byte je vždy znak "?", tzn. hodnota 0x3F na základě ASCII. Druhý byte má předem definované složení viz. tab. č. 21. V rámci jsou obsaženy základní informace o aktuálním stavu ST7580. Status rámec je vždy posílán od ST7580 k zařízení na HI jako potvrzení, že bylo externímu kontroleru dovoleno zahájit vysílání.

Tab. 21: Status rámec [9].

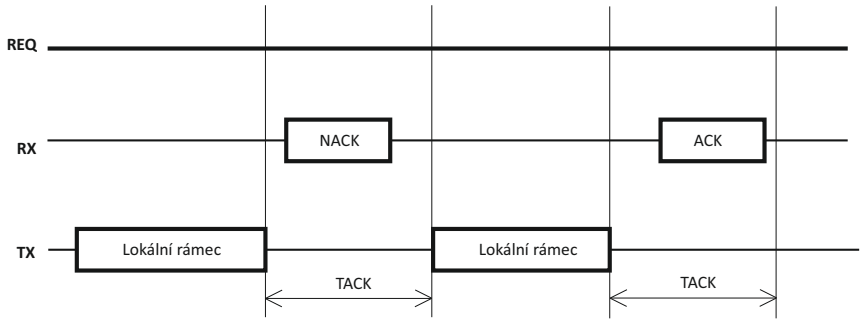
Byte	Bit	Popis	Význam nastavení
0	—	První byte	vždy 0x3F
1	0	Stav po rekonfiguraci	0: rekonfigurace se vykonala validně
			1: rekonfigurace se vykonala nevalidně
	1	Stav vysílání	0: ST7580 právě nevysílá PLC data
			1: ST7580 právě vysílá PLC data
	2	Stav příjmu	0: ST7580 právě nepřijímá PLC data
			1: ST7580 právě přijímá PLC data
	3–4	Aktuálně aktivní vrstva (z pohledu PLC)	0: PHY
			1: DL
			2: SS
			3: ST7580 není nakonfigurováno
	5	Nadproudové hlášení	0: žádný nadproudový stav
			1: alespoň jeden nadproudový stav
	6–7	Teplota čipu ST7580	0: T<70
			1: 70<T<100
			2: 100<T<125
			3: T>125

4.2.5 Příklad komunikace z master do slave

Pokud chce ST7580 vyslat data, nemusí vyžadovat komunikaci a data ihned odešle – obr. č. 18. Pole STX bude 0x02. Pokud je kontrolní součet správný a pole Length v LOC odpovídá velikosti délky pole Data, je MCU posláno ACK. Pokud dojde v procesu k:

- nekorektní příjem slave zařízením (NACK),
- kolizi s dalším vysíláním,
- nepotvrzení v rámci TACK,

ST7580 opakuje přenos a to pouze 1× po uplynutí TACK, ale pole STX je nyní 0x03. Při opětovném posílání se již potvrzení (ACK i NACK) neposílá (předpokládá se ACK).

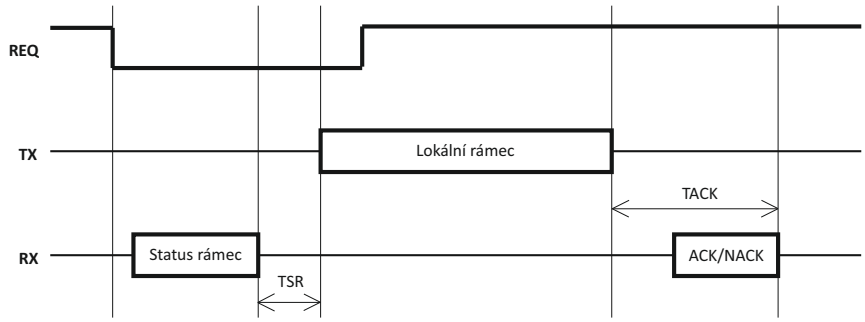


Obr. 18: Znázornění principu vyslání rámce z master do slave.

4.2.6 Příklad komunikace ze slave do master

Nyní je popsán průběh komunikace ve směru od slave (MCU) do master (ST7580) po silnoproudém vedení, viz obr. č. 19. Již bylo popsáno, že pokud chce slave vysílat, musí z výchozí hodnoty, která je na pinu REQ (logická 1), přejít do stavu logické 0. Pokud není ST7580 zaneprázdněn, odpoví Status zprávou potvrzující, že je kanál volný. Pak je MCU vyslán Lokální rámec s hodnotou pole STX nastaveného na 0x02. Vše musí proběhnout v časovém intervalu TSR. Pin REQ musí být ihned, jak je posláno pole STX Lokálního rámce, nastaven zpět na logickou 1.

Pokud ST7580 nepřijme první byte rámce v rámci časového intervalu TSR, je celý rámec ignorován a zahozen. Pokud sedí časové intervaly pro synchronizaci a je platný kontrolní součet a délka rámce, je masterem odesláno potvrzení ACK, ve všech ostatních případech je odesláno NACK. Potvrzení musí být přijato v rámci dalšího časového intervalu TACK, jinak se rámec považuje za nepotvrzený a celý vysílací proces se musí opakovat a to i v případě NACK. Proces opakování přenosu viz výše (změna STX).



Obr. 19: Znázornění principu vyslání rámce ze slave do master.

4.2.7 Příkazové kódy

Jedná se o hodnoty, kterých může nabývat pole Příkazových kódů o délce 1 B obsaženo v Lokálním rámci. Na základě tohoto pole master a slave zařízení zjistí, o jaký typ Lokálního rámce se jedná. Dále se budou označovat zkratkou CC (Command Codes). Jsou definovány čtyři hlavní skupiny CC z pohledu účelu:

1. **Žádosti** – vysílá je pouze externí MCU za účelem vyžádání služby od ST7580,
2. **Potvrzení** – vysílá je pouze ST7580 jako odpověď MCU, že předchozí CC Žádosti byla korektně vykonána,
3. **Chyby** – vysílá je pouze ST7580 jako odpověď MCU, že předchozí CC Žádosti nebyl vykonán,
4. **Indikace** – vysílá je pouze ST7580 MCU, že došlo k resetu čipu ST7580 nebo byly přijaty PLC data.

V oficiálním manuálu [9] jsou uvedeny všechny čtyři skupiny detailně. Dalším způsobem rozlišení je podle prvního sloupce Skupina, do které CC spadají a těmito Skupinami jsou:

1. **Reset** – SW reset čipu ST7580,
2. **MIB** – požadavek na operaci s MIB objektem,
3. **Ping** – ping mezi MCU a ST7580,
4. **Data** – operaci s daty určenými pro PLC,
5. **Chybové** – detekována určitá chyba.

4.2.8 Podoba rámce pro jednotlivé CC

Rámec bude odpovídat zmíněnému Lokálnímu rámci a bude se měnit obsah pole Data. Podoby pole Data pro jednotlivé typy CC jsou přístupné z uživatelské příručky ze zdroje [9] strana 19–33. Tyto tabulky zcela popisují, jak bude rámec vypadat. Dále je zde také proveden výčet hexadecimálních kódů pro jednotlivé CC.

4.3 Komunikační protokol MCU-PC

Poslední uvedený komunikační protokol je implementován mezi MCU a nadřazeným systémem (PC). Funkce P03 je, že se jedná zejména o dotazovací protokol, což odpovídá požadavkům na nadřazený systém, který má sloužit primárně jako monitorovací.

Návrh protokolu je celý vlastní a jeho části nejsou přebírány. Tento protokol nemá prioritu v SW implementaci MCU, vždy bude mít prioritu v MCU řešení samotné PLC komunikace. Komunikaci přicházející od PC bude MCU zapouzdřovat do příslušných PLC rámců a posílat na PLC síť. MCU má svrchované právo, zda žádosti od nadřazeného systému vyhoví. Rychlost komunikace bude 115200 Bd.

4.3.1 Složení rámce

Rámec protokolu P03 je zobrazen na obr. č. 4.3.1. Skládá se ze 4 polí a může nabývat velikosti 4–259 B. Význam jednotlivých polí je následující:

- **FSB** – Frame Start Byte; označuje začátek nového rámce; může nabývat tří hodnot:
  - 0xAA – první přenos konkrétního rámce,
  - 0xAF – opakovaný přenos konkrétního rámce,
  - 0xAC – nekompletní rámec; následuje rámec, jehož Data navazují na předchozí,
- **FT** – Frame Type; označuje typ přenášených dat, respektive určení rámce,
- **Length** – celková bytová délka pole Data,
- **Data** – přenášená data.

1 B	1 B	1 B	1 B až 255 B
FSB	**FT**	**Length**	**Data**
4 B až 258 B			

Obr. 20: Rámec protokolu P03

4.3.2 Potvrzování komunikace

Potvrzuje se každé přijetí rámce. Zpráva bude mít velikost 1 B a bude nabývat hodnot 0x33 pro potvrzení bezchybného příjmu (`ACK`) a 0x44 pro potvrzení chybového přenosu (`NACK`) nebo přijetí rámce, který nedává v daném kontextu komunikace smysl. Pokud je vysláno `NACK`, druhý účastník komunikaci přenos opakuje a to pouze jednou a se změněnou hodnotou PRE pole.

Typy rámců

Typ rámce určuje příjemce podle FT. Na základě typu rámce se mění skladba pole Data. V tab. č. 22 jsou uvedeny všechny navržené typy rámců.

Tab. 22: Typy rámců protokolu P03

Označení	Hodnota	Odesílatel	Možná odpověď
AUTO_REPORT	0x11	MCU	ACK NACK
CONFIG_REQ	0x22	PC/MCU	CONFIG_REQ
DATA_REQ	0x33	PC/MCU	DATA_REQ
PING_REQ	0x44	PC/MCU	PING_REQ
DEV_TABLE_REQ	0xCC	PC/MCU	DEV_TABLE_REQ
RECONFIG_REQ	0xE0	PC/MCU	RECONFIG_REQ
GENERAL_REQ	0x90	PC/MCU	GENERAL_REQ

Důležitým faktem je, že možnou odpovědí na většinu typů rámců je stejný rámec. Tohoto lze využít, jelikož není důvod aby MCU odesílalo PC žádost o např. tabulku zařízení. Takový rámec poslaný od MCU se tedy bere jako odpověď na původní žádost od PC. Popis jednotlivých FT kódů:

1. **`AUTO_REPORT`** – tato zpráva je odesílána MD MCU vždy, když MD odesílá nebo přijímá PLC rámec; v poli Data se z MCU exportuje celý rámec; odpovědí jsou pouze zprávy potvrzení `ACK`, `NACK`,
2. **`CONFIG_REQ`** – žádost nadřazeného systému o zaslání aktuální konfigurace konkrétního zařízení; pole data má velikost 1 B a jedná se o adresu příslušného zařízení; odpovědí je tentýž typ rámce, ale nyní jsou od 2. bytu v poli Data obsažena příslušná žádaná data,
3. **`DATA_REQ`** – žádost o určitá data od konkrétního zařízení; pole Data obsahuje adresu příslušného zařízení, tzn. má mít velikost 1 B; odpovědí je tentýž typ rámce, ale nyní jsou od 2. bytu v poli Data, obsažena příslušná žádaná data; formát těchto dat závisí na typu zařízení,
4. **`PING_REQ`** – ping z nadřazeného systému na adresu obsaženou v 1. bytu pole Data; pole Data má velikost 1B; odpovědí je tatáž zpráva,

5. `DEV_TABLE_REQ` – žádost od PC o zaslání kompletní tabulky zařízení připojených v síti; odpovědí je tatáž zpráva v Datech obsahující tabulku zařízení,
6. `RECONFIG_REQ` – žádost o rekonfiguraci příslušného zařízení, jehož adresa je obsažena v 1. bytu pole Data; ostatní byty Data pole je požadovaná konfigurace; tato data mají složení odpovídající typu zařízení; odpovědí je tatáž zpráva, která má velikost pole Data 2 B, z nichž první je adresa příslušného zařízení a druhý je kód `ACK/NACK`, analogicky k tomu, zda rekonfigurace proběhla úspěšně,
7. `GENERAL_REQ` – žádost, je specifikována v poli Data, respektive v poli Data je obsažen přímo NL rámec protokolu P01, který má být odeslán od MD.

4.3.3 Více-rámcové zprávy

Hodnota pole PRE může nabývat i hodnoty `0xAC`. Tato hodnota značí, že příslušný tok rámců na sebe navazuje, respektive pole Data na sebe navazují.

Jako ukončení této sekvence rámců je posíláno `ACK` potvrzení – tzv. **ukončující ACK**. Tato technika funguje, jelikož druhá strana nečeká žádné `ACK/NACK` zprávy od odesílatele, takže je lze takto použít. Pokud se chybně přijme i jeden ze sekvence rámců, celá sekvence se opakuje a to pouze jednou. Každý rámec sekvence se nepotvrzuje, ale posílá se potvrzení o příjmu na konci sekvence, po příjmu ukončovacího `ACK`.

5 POPIS FIRMWARU PRO KONTROLÉR

V této kapitole je rozebrána praktická SW implementace v MCU jednotce (MD a SD zařízení). Tyto implementace vycházejí z navržených komunikačních protokolů. Není zde uveden celý kód, ale jen jeho blokové zobrazení s vysvětlením toku programu z pohledu MD a SD zařízení a následně popis vytvořených funkcí (jen hlavičky funkcí) a globálních proměnných (tzn. lokální a pomocné proměnné jsou z pochopitelných důvodů vynechány).

5.1 Princip uchovávání dat v zařízení

Zde je rozebrán popis uchovávání dat v každém jednotlivém účastníkovy komunikace v AS01. K tomuto jednotnému způsobu je přikročeno zejména kvůli zobecnění komunikace po PLC.

Každé zařízení (MD i SD) v sobě uchovává informace o svém aktuálním nastavení. Jedná se o strukturu s názvem `"Device"`. Tato struktura obsahuje proměnné uvedené v tab. č. 23. Popis skladby struktury:

- **Address** – adresa zařízení
- **MS** – zkratka Master/Slave; určuje, zda je zařízení MD nebo SD
- **RW** – zkratka Read/Write; určení, zda je zařízení určeno ke čtení/zapisování
- **DeviceType** – kód typu zařízení; viz podkapitola č. 3.1.2
- **Active** – zařízení je aktivní nebo neaktivní (např. dočasně odpojeno)
- **LastValue** – aktuální hodnota z připojeného přídavného modulu (jen SD zařízení); v MD je nastaveno na 0.0f; lze nastavit 3 pozice na levé straně od desetinné tečky a 3 pozice na pravé straně od desetinné tečky
- **Config** – pole uchovávající aktuální konfiguraci **přídavného modulu**
- **Registrating** – určuje zda, zařízení aktuálně není v procesu registrace do sítě

Tab. 23: Struktura uchovávání dat ve slave a master.

Proměnná	**Datový typ**
`Address`	uint8_t
`MS`	uint8_t
`RW`	uint8_t
`DeviceType`	uint8_t
`Active`	uint8_t
`LastValue`	float
`Config`	uint8_t[8]
`Registrating`	uint8_t

5.2 Popis logického toku programu

V této kapitole je popsán logický tok programu pro obě verze firmware – MD a SD jednotky. Tyto verze jsou tvořeny programovým vybavením zmíněným v následující kapitole č. 5.3.

Procesový přístup

Verze firmware pro MD zařízení obsahuje jednoduchý procesový systém. Tento systém bylo nutné implementovat z důvodu kontinuity komunikace. Jednoduše řečeno, MD zařízení si musí pamatovat, jaký rámec se od určitého SD očekává jako odpověď na rámec k němu odeslaný. Jinak nebude zachována kontinuita komunikace na úrovni DL. Respektive např. potvrzovací zprávy ztratí význam, jelikož MD nebude vědět, proč přicházejí. Tato funkcionalita je zajištěna prvky:

- **struktura `RunningProcess`** – informace o právě jednom procesu; obsahuje informace o tom, jaký typ rámce se očekává a od koho a zda je proces aktivní,
- **pole struktur `processesTab`** – složeno ze struktur `RunningProcess`; uloženy všechny aktivní/neaktivní procesy (neaktivní obsahuje do té doby, dokud nedojde k jejich přepsání procesem novým); jako ID procesu je využita poloha v tomto poli,
- **funkce `registrateNewProcess`** – registrace nového procesu,
- **funkce `lookForProcess`** – prohledání pole `processesTab`; pokud najde odpovídající proces, vrátí ukazatel na proces,
- **funkce `cleanUpProccessTab`** – pokud je pole `processesTab` plně naplněno aktivními procesy, je přikročeno ke spuštění této funkce, která řeší, které procesy je možné vymazat, respektive je deaktivovat.

Nutno podotknout, že tato funkcionalita je implementována pouze v master, jelikož z principu sítě AS01, slave pouze odpovídá na dotazy od master. Takže zde není důvod implementace ve slave.

5.2.1 Jednotka Master Device

Blokové znázornění toku programu v master je zobrazeno na obr. č. 21. Nyní je logický tok firmwaru v MD zjednodušeně popsán.

Hned po připojení k napájení proběhne inicializace potřebných periferií, časovačů a přerušení. Po-té následuje inicializace ST7580 (funkce `master_INIT`). Tato inicializace je nastavením příslušných MIB tabulek na základě navržených komunikačních protokolů a parametrů přenosu pro AS01. Nakonec jsou všechny globálně použité proměnné nastaveny na své výchozí hodnoty.

Následně program vstupuje do nekonečné smyčky (funkce `master_loop`). Nejprve se načte poslední indikovaný rámec z ST7580 (funkce receptionPLC[1]) – konfigurace pro export dat PLC sítě čipem ST7580, je nastavena tak, že se exportuje pouze pole Data DL rámce, což odpovídá rámci NL vrstvy protokolu P01. Tento rámec se verifikuje ve funkci `plcFrameVerification` z pohledu:

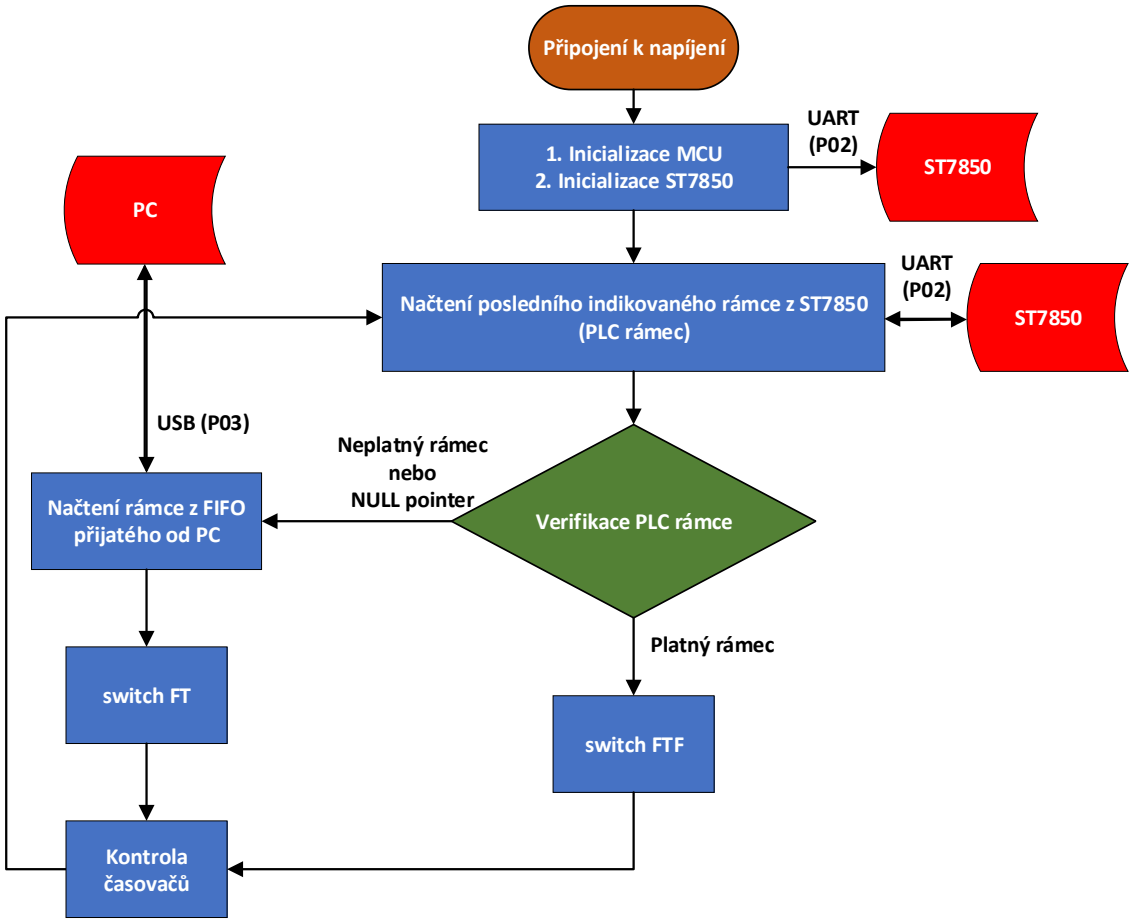
[1] Jedná se o ukazatel na rámec.

1. zda je zdrojová adresa obsažena v tabulce zařízení (označení `slaveDevTab`) – pokud není, je volána funkce `slave_address_violation`, která zajistí reset SD zařízení, které se po resetu pokusí registrovat do AS01,
2. zda délka exportovaného DL rámce odpovídá hodnotě v poli Length na DL úrovni.

Nutno podotknout, že cílová adresa je kontrolována hned při příjmu rámce ve zmíněné funkci `receptionPLC` (funcke `NET_frame_verification`). Tzn. pokud cílová adresa neodpovídá zařízení, je rámec zahozen.

Pokud verifikační funkce vyhodnotí rámec jako validní, je přikročeno do hlavního stavového automatu (`switch`), kde se provede odpovídající příkaz na základě pole FTF (DL rámec). Pokud verifikace vyhodnotí rámec jako neplatný či je z funkce `receptionPLC` vrácen `NULL` ukazatel, je přikročeno k načtení rámce přijatého z nadřazeného systému (funkce `externComFrameProcess`).

Celá hlavní smyčka je ukončena kontrolou časovačů. Jde o časové intervaly popsané v podkapitole č. 11, které zajišťují synchronizaci komunikace na úrovni DL vrstvy.

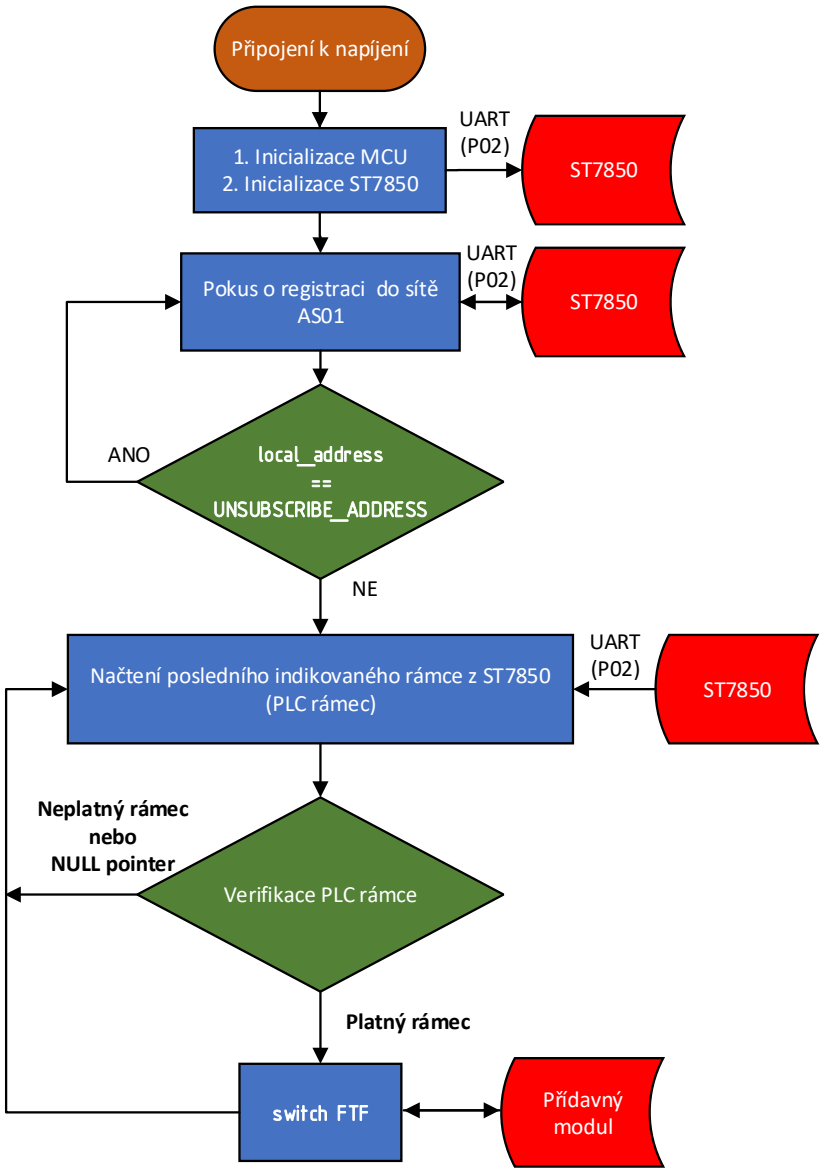


Obr. 21: BS firmware verze pro MD a logický tok programu.

5.2.2 Jednotka Slave Device

Na obr. č. 22 je uvedené grafické znázornění BS a jeho programového toku. Hlavní smyčka `slave_loop` plní stejné úkony jako u MD, ale neodbočuje k řešení rámců od nadřazeného systému, jelikož SD nebude k němu přímo připojen. Naopak navíc se zde nachází registrace do sítě ihned po inicializaci všech periferií a ST7580.

Registrace probíhá stejně jako standardní PLC komunikace na úrovni DL vrstvy AS01. Pouze slave vystupuje pod adresou `NON_SUBS_ADDRESS`, díky které master pozná, že se v síti objevil slave, který žádá o registraci. Funkce `subs_request` je volána v nekočné smyčce, dokud nedostane přidělenou adresu od master (je zde určitý výše posaný časový interval TTR, aby nedošlo k zahlcení sítě).



Obr. 22: Diagram SW řešení SD zařízení.

5.3 Programové vybavení

Níže v této kapitole je popsáno programové vybavení zajišťující vykonávání algoritmu uvedeného v kapitole č. 5.2.1 pro MD a v kapitole č. 5.2.2 pro SD.

5.3.1 Převzaté programové vybavení

Do této kategorie spadají knihovny a ovladače používané pro vytváření vlastního programového vybavení. Na těchto částech je v celém SW řešení stavěno. Tzn. nejsou vytvořeny jako vlastní výstup této práce. Jedná se o dvě skupiny knihoven:

1. **HAL knihovny** – Hardware Abstraction Layer; jedná se o soubor knihoven vytvořených programem CubeMX, tak jak je uvedeno v podkapitole č. 1.2.1,
2. **ST7580 ovladač** – výrobcem vytvořený oficiální ovladač pro komunikaci s ST7580; je dostupný z [11]; jde o SW implementaci protokolu P02.

5.3.2 Vytvořené programové vybavení

Zde je uveden výčet souborů vytvořených jako vlastní řešení této práce. Z těchto souborů jsou tvořeny jednotlivé verze firmware pro MD a SD, respektive konkrétní typ SD na základě přídavného modulu (základní činnost zařízení, respektive implementace P01).

Popis programového vybavení je omezen na vlastní přínos práce a použité ovladače či knihovny se nerozebírají. Níže jsou uvedeny vlastní vytvořené soubory a stručný popis:

- **A) net_config.h** – konstanty a proměnné použité pro PLC síť, tzn. protokol P01,
- **B) PLC_communication.h & PLC_communication.c** – zajišťují PLC komunikaci,
- **C) slave.h & slave.c** – soubor funkcí používaných SD zařízením,
- **D) master.h & master.c** – soubor funkcí používaných MD zařízením,
- **E) slaveAPP.h & slaveAPP.c** – soubory hlavní smyčky programu pro SD,
- **F) masterAPP.h & masterAPP.c** – soubory hlavní smyčky programu pro MD.

5.3.3 Společné programové vybavení

Z těchto částí vychází firmware pro SD i MD. Jedná se zejména o funkce a metody zajišťující pravidla PLC komunikace. Dále sem patří konfigurační konstanty, kódy pro jednotlivá pole rámců NL vrstvy P01 a pro P03 a definice globálních konstant/proměnných.

A) Soubor net_config.h

Tento hlavičkový soubor obsahuje výčet všech konstant a globálních proměnných používaných v NL vrstvě protokolu P01. Jde o FTF kódy, globální proměnné a ostatní konstanty používané v každém z obou verzí firmware. FTF kódy a ostatní konstanty uvedené v popisu protokolu P01 se zde neuvádějí, jelikož jsou již uvedeny v podkapitole č. 4.1.

Níže je uveden výčet globálních proměnných/konstant zde deklarovaných a jejich datový typ, případně i výchozí hodnota a v tab. č. 24 je uveden jejich popis. Jedná se tedy o konstanty nedeklarované v popisu P01 v podkapitole č. 4.1 – tato část je řešena preprocesorovým direktivem **`#define`**. V souhrnu jsou zde tedy obsaženy:

- **Kódy FTF** (P01),
- **Adresní prostor na základě požadavků AS01**,
- **Fixní parametry PLC komunikace**,
- **Hodnota časových intervalů pro P01**.

Proměnné a konstanty:

```
#define GENERAL_VALUE      0x38
#define TRUE               0x39
#define FALSE              0x3A

#define MASTER             FALSE
#define SLAVE              TRUE

#define READ_DEVICE        FALSE
#define WRITE_DEVICE       TRUE

#define SENSOR_BOOL_R      0xA1
#define SENSOR_BOOL_RW     0xA2
#define SENSOR_FLOAT_R     0xB1
#define SENSOR_FLOAT_RW    0xB2

#define LED_SLAVE_BLINK    0xFA
#define SET_VALUE          0xFB

typedef uint8_t ubyte8;
extern ubyte8 token;

static const uint8_t modem_configAPP[1] = { 0x11 };
static const uint8_t phy_configAPP[14] = { 0x01, 0x11, 0x70, 0x01, 0x5F,
                                           0x90, 0x0F, 0x15, 0x00, 0x00,
                                           0x02, 0x35, 0x9B, 0x58       };
static const uint8_t ss_configAPP[16] = { 0xAA, 0xAA, 0xAA, 0xAA, 0xAA,
                                          0xAA, 0xAA, 0xAA, 0xAA, 0xAA,
                                          0xAA, 0xAA, 0xAA, 0xAA, 0xAA,
                                          0xAA                        };

typedef struct
{
  uint8_t destination;
  uint8_t frame_type;
  uint8_t source;
  uint8_t frame_length;
  uint8_t data[252];
} PLCframe;
```

```
39 typedef struct
40 {
41   ubyte8 Address;
42   ubyte8 MS;
43   ubyte8 RW;
44   ubyte8 DeviceType;
45   ubyte8 Active;
46   float  LastValue;
47   ubyte8 Config[8];
48   ubyte8 Registrating;
49 }Device;
50
51 typedef struct
52 {
53   ubyte8 whatExpecting;
54   ubyte8 fromWho;
55   ubyte8 active;
56 }RunningProcess;
```

Tab. 24: Proměnné a konstanty deklarované v souboru net_config.h.

Proměnná / konstanta	**Popis**
`GENERAL_VALUE`	Výchozí hodnota pro 1 B (výchozí inicializace proměnných) V podstatě zastupuje hodnotu `NULL`
`TRUE` `FALSE`	Náhrada za jednobitové proměnné typu bool
`MASTER` `SLAVE`	Kód typu koncového PLC modulu
`READ_DEVICE` `WRITE_DEVICE`	Kód zda je zařízení určeno pro čtení či i zápis

`SENSOR_BOOL_R`
`SENSOR_BOOL_RW`
`SENSOR_FLOAT_R`
`SENSOR_FLOAT_RW`
	Kódy typů zařízení

`LED_SLAVE_BLINK`	Kód příkazu pro bliknutí diody v SD
`SET_VALUE`	Kód příkazu pro nastavení LastValue v SD
`ubyte8`	Definice datového typu `uint8_t`
`token`	Označení, zda má příslušné zařízení token
`modem_configAPP`	Konfigurace 0x00 MIB tabulky
`phy_configAPP`	Konfigurace 0x01 MIB tabulky
`ss_configAPP`	Šifrovací klíč umístěný v MIB tabulce 0x02
`PLCframe`	Předloha pro obecný tvar rámce na úrovni NL vrstvy P01
`Device`	viz podkapitola č. 5.1
`RunningProcess`	Struktura; jeden proces

B) Soubory PLC_communication.h & PLC_communication.c

Jedná se o soubory zajišťující pravidla PLC komunikace, která jsou společná pro všechny účastníky komunikace. Níže je uveden výčet všech součástí a jim odpovídající tabulky s popisem významu – proměnné a konstanty v tab. č. 25 a funkce viz tab. č. 26.

Proměnné a konstanty:

```
PLCframe PLCframeAPP;
uint8_t  rawOutput[256];
```

Tab. 25: Proměnné a konstanty deklarované v souborech PLC_communication.h a PLC_communication.c.

Proměnná/Konstanta	Popis
`PLCframeAPP`	Rámec NL vrstvy protokolu P01
`rawOutput`	Surová data určená k PLC vysílání

Deklarované funkce:

```
ubyte8 send_ack(ubyte8 channel, ubyte8 ack_type, ubyte8 dest_address,
                ubyte8 frame_type_ack);
ubyte8* NET_frame_builder(ubyte8 address, ubyte8 frame_type,
                          ubyte8* data, ubyte8 data_length);
PLCframe* receptionPLC(void);
ubyte8 NET_frame_verification(PLCframe* frame);
```

Tab. 26: Funkce implementované v souborech PLC_communication.h a PLC_communication.c.

Funkce	Popis
`send_ack`	Zaslání potvrzení
`NET_frame_builder`	Funkce vracející sestavený rámec NL vrstvy AS01
`receptionPLC`	Funkce vracející přijatý PLC rámec
`NET_frame_verification`	Funkce kontrolující přijatý PLC rámec na úrovni NL vrstvy

5.3.4 Programové vybavení MD zařízení

Zde je popsáno programové vybavení pro MD verzi firmware. Tento firmware je rozsahově pochopitelně rozsáhlejší než SD firmware.

C) Soubory master.h & master.c

Co se týče konstant/proměnných jedná se pouze o použití v P03 (např. FT kódy) – proto zde uvedeny nejsou. Pomocné funkce uvedené v tab. č. 27 zajišťují běh MD zařízení.

Deklarované funkce:

```
void ping_response_process(PLCframe* incomingFrame);
void ping(ubyte8 dest_address);
void new_subs_request(PLCframe* frame, Device* slaveDevice,
                      ubyte8 address_offer);
void upper_comm(char* dataToReport, int dataToReport_length);
void complete_subscription(PLCframe* frame, Device* slaveDevice);
void data_read(ubyte8 dest_address, ubyte8* dataToRead,
               ubyte8 dataToRead_length);
void data_write(ubyte8 dest_address, ubyte8* dataToWrite,
                ubyte8 dataToWrite_length);
void processIncommingData(PLCframe* frame, Device* slaveDev);
void slave_address_violation(ubyte8 address);
void slave_config_read(ubyte8 dest_address);
void slave_config_write(ubyte8 dest_address, ubyte8 *newConfig,
                        ubyte8 newConfig_length);
ubyte8 send_token(ubyte8 dest_address);
ubyte8 token_force_pull(ubyte8 dest_address);
ubyte8 high_priority_event(PLCframe* incomingFrame, ubyte8 eventType);
```

Tab. 27: Funkce implementované v souborech master.h a master.c.

Funkce	**Popis**
`ping_response_process`	Po žádosti `PING_REQUEST` přijde odpověď `PING_RESPONSE` a tato funkce je právě tehdy volána
`ping`	`PING_REQUEST` vůči konkrétnímu SD (od MD)
`new_subs_request`	Zahájení procesu registrace nového slave do sítě na základě žádosti od konkrétního slave
`upper_comm`	Komunikace s nadřazeným systémem
`complete_subscription`	Dokončení procesu registrace nového slave zařízení do sítě na základě potvrzení o příjmu návrhu adresy od MD
`data_read`	Čtení dat z konkrétního SD zařízení
`data_write`	Zápis dat do konkrétního SD zařízení
`processIncommingData`	Voláno na základě příjmu rámce od NS
`slave_address_violation`	Vykonání kroků k zajištění nápravy při zjištění chybně registrovaného zařízení
`slave_config_write`	Vyžadovaná rekonfigurace SD zařízení
`slave_config_read`	Výčet kompletní aktuální konfigurace SD zařízení
`send_token`	Zaslání zprávy `TOKEN_TOKEN`
`token_force_pull`	Odebrání tokenu příslušnému SD zařízení
`high_priority_event`	Vykonání ošetření při výskytu události s nejvyšší prioritou

D) Soubory masterAPP.h & masterAPP.c

Zde je obsažena funkce hlavní nekonečné smyčky, jejíž blokové znázornění a vysvětlení je uvedeno v podkapitole č. 5.2.1. Dále se zde nachází inicializační funkce, která je volána po připojení zařízení k napájení. Obě tyto funkce viz tab. č. 29.

Proměnné a konstanty:

```
Device localHost;
ubyte8 token = TRUE;
ubyte8 token_owner = MASTER_ADDRESS;
short slaveCount = 0;
PLCframe* PLCframeAPPfinal;
Device slaveDevTab[MAX_SLAVE_DEVICES];
RunningProcess processesTab[MAX_ACTIVE_PROCESSES];
```

Tab. 28: Proměnné a konstanty deklarované v souborech masterAPP.h a masterAPP.c.

Funkce	Popis
`localHost`	Struktura uchovávající aktuální nastavení zařízení
`token`	Označuje, zda zařízení vlastní token
`token_owner`	Adresa zařízení aktuálně vlastnící token
`slaveCount`	Počet připojených SD zařízení
`PLCframeAPPfinal`	Ukazatel na poslední přijatý PLC rámec
`slaveDevTab`	Tabulka všech SD zařízení
`processesTab`	Tabulka aktuálně aktivních procesů

Deklarované funkce:

```
void master_INIT(void);
void master_loop(void);
ubyte8 registrateNewProcess(ubyte8 whatExpecting, ubyte8 fromWho);
void cleanUpProccessTab(void);
void cleanUpAddressTab (void);
RunningProcess* lookForProcess(ubyte8 address, ubyte8 frameType);
void deleteProcess(RunningProcess* p);
ubyte8 plcFrameVerification (PLCframe* frame);
void externComFrameProcess(ubyte8* commandFrame);
void initVars(void);
```

Tab. 29: Funkce implementované v souborech masterAPP.h a masterAPP.c.

Funkce	Popis
`master_INIT`	Inicializace při připojení MD k napájení
`master_loop`	Hlavní nekonečná smyčka MD zařízení
`registrateNewProcess`	Registrace nového procesu
`cleanUpProccessTab`	Při zaplnění tab. procesů se provede čištění tabulky
`cleanUpAddressTab`	Při zaplnění tab. zařízení se provede čištění tabulky
`lookForProcess`	Vyhledání procesu v tabulce
`deleteProcess`	Výmaz konkrítního procesu
`plcFrameVerification`	Kompletní kontrola NL rámce P01
`externComFrameProcess`	Vyhodnocení přijatého rámce P03
`initVars`	Inicializace všech proměnných

5.3.5 Programové vybavení pro SD zařízení

Níže jsou popsány soubory, ze kterých je tvořen obecný firmware pro SD. Verze firmware pro každé SD se mírně liší v typu připojeného přídavného modulu. Rozlišení typu SD, popsaného v podkapitole č. 3.1.2, se provádí pomocí preprocesorových direktiv. Tímto způsobem odpadá nutnost uchovávat několik zdrojových souborů pro rozdílné druhy SD.

E) Soubory slave.h & slave.c

Obsahují obecné funkce sloužící k činnosti SD zařízení. Výčet funkcí viz tab. č. 30.

Deklarované funkce:

```
void ping_response_process(ubyte8 source_address);
void data_send(ubyte8 dest_address, ubyte8 dataType, ubyte8* dataToWrite,
               int dataToWrite_length);
ubyte8 subs_request(PLCframe* frame);
ubyte8 reconfig_device(ubyte8* newConfig);
ubyte8 high_priority_event(ubyte8 source_address, ubyte8 eventType);
void token_request(void);
void token_return(void);
```

Tab. 30: Funkce implementované v souborech slave.h a slave.c.

Funkce	Popis
`ping_response_process`	Odpověď na žádost o ping od MD
`data_send`	Zaslání vyžádaných dat
`subs_request`	Žádost o registraci do AS01
`reconfig_device`	Rekonfigurace SD zařízení
`high_priority_event`	Obsloužení události s nejvyšší prioritou
`token_request`	Žádost o token
`token_return`	Vrácení tokenu

F) Soubory slaveAPP.h & slaveAPP.c

Mají analogicky stejný význam jako masterAPP u MD. Implementované funkce tab. č. 32.

Proměnné a konstanty:

```
#define SENSOR_FLOAT_R_DEV //#define SENSOR_BOOL_RW
ubyte8 token = FALSE;
PLCframe* PLCframeAPPfinal;
Device localHost;
```

Tab. 31: Proměnné a konstanty deklarované v souborech slaveAPP.h a slaveAPP.c.

Proměnná/Konstanta	Popis
`SENSOR_FLOAT_R_DEV` `SENSOR_BOOL_RW`	viz podkapitola č. 3.1.2
`token`	Zda zařízení vlastní token
`PLCframeAPPfinal`	Ukazatel na poslední přijatý P01 NL rámec
`localHost`	Struktura obsahující celkové nastavení SD zařízení

Deklarované funkce:

```
void slave_INIT(void);
void slave_loop(void);
void initVars(void);
```

Tab. 32: Funkce implementované v souborech slaveAPP.h a slaveAPP.c.

Funkce	Popis
`slave_INIT`	Funkce provádějící inicializaci po připojení SD zařízení k napájení
`slave_loop`	Hlavní smyčka SD zařízení
`initVars`	Inicializace použitých globálních proměnných

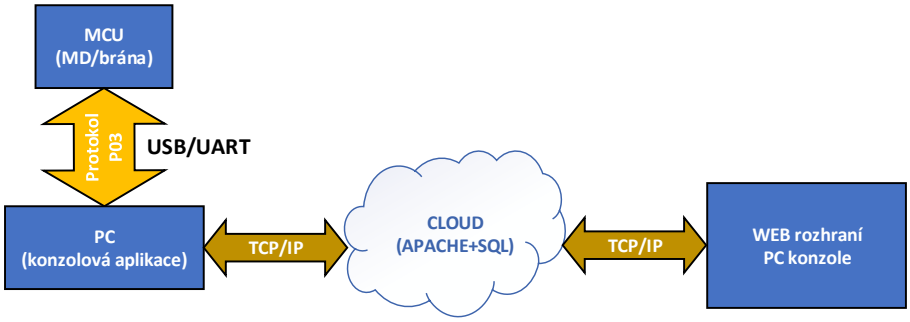
6 ŘEŠENÍ NADŘAZENÉHO SYSTÉMU

V této kapitole je popsáno celkové řešení nadřazeného systému včetně popisu vytvořeného programového vybavení a algoritmů.

6.1 Popis řešení

Řešení nadřazeného systému (zkráceně NS) je zobrazeno na obr č. 23 a vycházní z hlavního BS, viz obr. č. 5, kde je řešení naznačeno obecně.

Jak bylo popsáno v předchozích kapitolách, MD komunikuje s NS pomocí USB (protokol P03). V PC je spuštěna konzolová aplikace, která rozčleňuje přijatá data a odesílá je na server, respektive do SQL databáze umístěné na tomto serveru. Server je řešen pomocí distribuce Apache. K serveru se lze připojit pomocí webového rozhraní nebo pomocí PC konzole. Podrobněji je popis řešení uveden v následujících podkapitolách.



Obr. 23: Blokové schéma řešení nadřazeného systému.

6.2 Konzolová PC aplikace

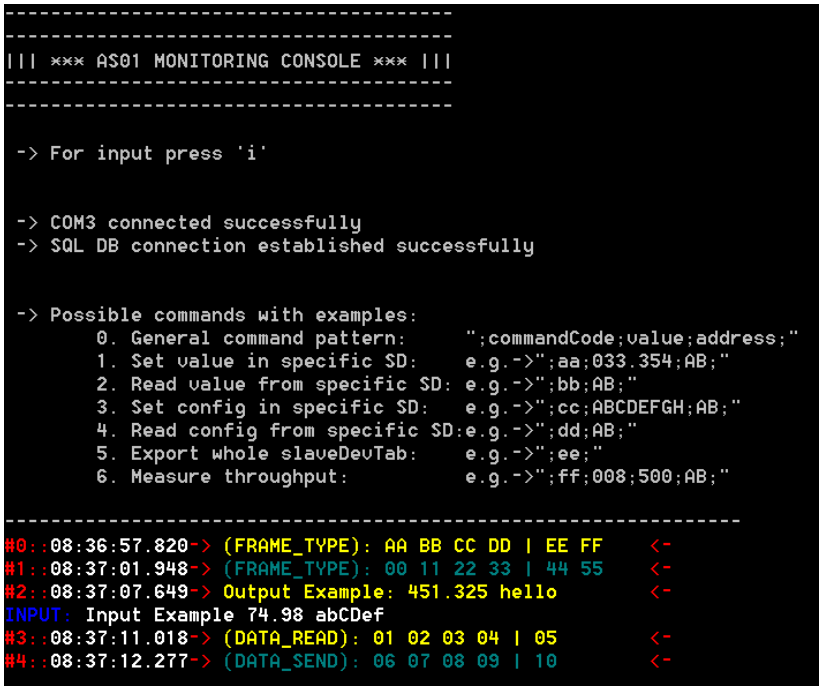
Tato aplikace zachytává data od MD, následně je člení a posílá pomocí TCP/IP na server (SQL databáze). V rámci aplikace je implementována konzole, která se používá pro zobrazení zasílaných dat od MD. Tato data jsou formátována do uživatelsky přívětivé podoby, opatřena časovým razítkem a zobrazena. Pro účely právě je nadále označována jako KA77 (Konzolová aplikace 77).

KA77 je napsána v jazyce C#. Snímky zachycující činnost aplikace viz kapitola č. 7 (zde je popsána aplikace v činnosti). Je použita jako testovací nástroj či lépe řečeno paketový analyzátor pro celkovou funkci sítě AS01.

V činnosti bude KA77 popsána v následujících kapitolách, zde se ale provede seznámení s formátem zobrazovaných dat (příklad viz obr. č. 24). Nyní je výše zmíněný obrázek analyzován:

- Nejprve je zobrazena hlavička programu KA77 s výpisem, zda došlo úspěšně k připojení k SQL DB serveru,
- následují řádky označené symbolem "#" a následným číslem řádku,

- po dvojnásobné dvojtečce je zobrazena časová značka, ve kterou byla daná událost přijata; je to čas příjmu aplikací KA77,
- uvnitř červených šipek se nachází samotná zobrazovaná informace,
- v kulatých závorkách se nachází název typu DL rámce (FTF); pokud je výpis bez závorek, jedná se pouze o informační výpis generovaný MD,
- za kulatými závorkami se nachází minimálně první 4 B (hlavička) DL rámce; maximálně je zobrazeno prvních 8 B,
- pokud je výpis (v červených šipkách) žluté barvy, jedná se o zprávu vysílanou MD; tato zpráva jde na silové vedení, pokud je ve výše zmíněných kulatých závorkách, jinak se jedná jen o informační výpis,
- pokud je výpis barvou tyrkysovou (v červených šipkách), jedná se o data přijímaná po silovém vedení MD zařízením od SD zařízení,
- po zmáčknutí klávesy `"i"`, je zpřístupněna možnost zaslání dat do MD – modrý výpis mezi řádky `#2` a `#3`.



Obr. 24: Formátování výpisu v konzolové aplikaci KA77.

6.2.1 Možnosti zaslání povelů do master

Po stlačení klávesy `"i"`, lze vložit text, který je odeslán do MD. Ten pak na základě přijatých dat (pokud jsou ve správném formátu) vykoná povel. Aby se nemusel vkládat přímo rámec P03, tak bylo určeno několik snadněji zapamatovatelných řetězců. Ty po vložení do konzole aplikace K77 rozčlení na rámec P03. Všechny povely viz tab. č. 33.

Z výše zmíněné tabulky vyplývá, že každý povel se skládá z několika částí, vzájemně oddělených středníkem. První část je příkaz (např. `"aa"`), ale ostatní části jsou proměnné.

Pravidlem je, že u příkazů, které něco nastavují, následuje hodnota, na kterou se má nastavit, a jako poslední část je adresa[1] koncového PLC zařízení v hexadecimální podobě.

Znovu je nutno zmínit, že při zadávání požadované hodnoty `LastValue` do konzole, **je nutné číslo zadat ve specifickém tvaru** – tři čísla nalevo od desetinné tečky a tři čísla napravo od desetinné tečky a to i v případě pokud se jedná o nuly! Taktéž požadovaná **konfigurace se vždy skládá právě a pouze z 8 B** !

Tab. 33: Povely pro ovládání MD z konzole KA77.

Povel	Popis
;aa;031.700;AB;	Nastavení hodnoty `LastValue` na 31.7 v SD s adresou 0xAB
;bb;AB;	Výčet hodnoty `LastValue` z SD s adresou 0xAB
;cc;ABCDEFGH;AB;	Nastavení `config` na "ABCDEFGH"v SD s adresou 0xAB
;dd;AB;	Výčet `config` z SD s adresou 0xAB
;ee;	Žádost o kompletní tabulku `slaveDevTab`
;ff;008;500;AB;	Měření propustnosti s velikostí pole Data 8 B, 500 opakování přenosu rámce na adresu 0xAB

6.3 Webový server a SQL databáze

Jako řešení této části bylo zvoleno použití balíčku od společnosti Apache Friends jménem XAMPP (zkratka pro Cross-Platform (X), Apache (A), MySQL (M), PHP (P) and Perl (P)), dostupné z [16]. Jak již rozbor zkratky napovídá, jedná se o ucelený balíček obsahující Apache server, SQL databázi a jazyky PHP a Perl. Z tohoto balíčku jsou použity následující funkcionality:

- **Apache server** – jakožto webový server obsahující html stránky webového rozhraní.
- **MySQL** – databáze obsahující konfiguraci a historii komunikace po síti AS01.
- **PHP** – programovací jazyk použitý pro psaní skriptů pro řešené webové rozhraní.

6.3.1 Databáze

Je použit SQL jazyk, respektive SQL databáze – viz oficiální stránky dostupné z [15]. Jedná se o standardizovaný strukturovaný dotazovací jazyk. Používá se právě pro práci s daty v relačních databázích.

Tabulka zařízení

Rozvržení tabulky zařízení odpovídá struktuře `Device` popsané v podkapitole č. 5.1. Tzn. ukládají se data o každém připojeném zařízení k síti AS01 v přesně takovém tvaru, jako je struktura `Device`. Ukázka rozvržení tabulky viz tab č. 34.

[1] Jedná se o adresu ve smyslu NL vrstvy P01.

Tab. 34: Tabulka zařízení.

Address	MD/SD device	R/W device	Device type	Active	Last value	Config	Is Registrating

Tabulka záznamů událostí

Tato tabulka je použita pro uchování historie komunikace na síti AS01. Respektive jedná se o informační zprávy odeslané z MD a následně zobrazené v KA77. Zaznamenávají se informace o každém zaslaném paketu. Následně je KA77 v PC rozčlení, přidá časovou známku (timestamp) a odešle do příslušné tabulky SQL databáze. Tato tabulka obsahuje:

- **Timestamp** – časové razítko; čas příjmu konzolovou aplikací KA77,
- **Who** – určuje, k jakému zařízení se informace vztahuje; 1 B adresa zařízení (NL vrstva AS01),
- **Event** – druh zprávy; již rozčleněná data převedená do textového řetězce (do uživatelsky přívětivé podoby).

Tab. 35: Tabulka záznamů událostí.

Timestamp	Who	Event

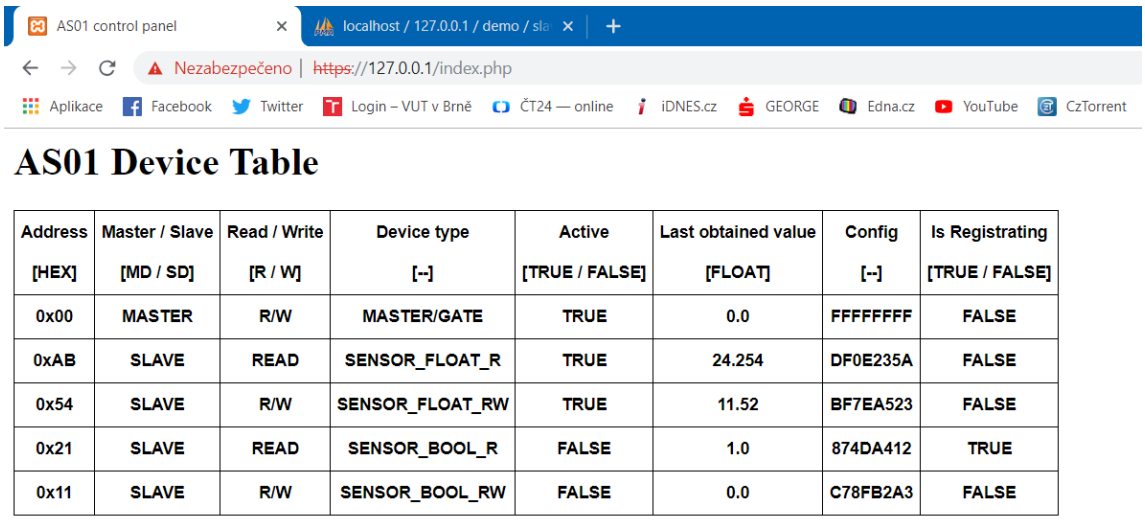
6.3.2 Webové rozhraní

Níže je uveden kód pro webové rozhraní. Jedná se o jednoduchou webovou stránku, kde je vypsána tabulka zařízení zmíněná výše. Tuto část zajišťuje PHP script na řádcích 16 až 44. Na obrázku č. 25 je zobrazena výsledná podoba[2].

```
1  <html>
2    <head><title>AS01 control panel</title></head>
3    <body>
4      <h1>AS01 Device Table</h1>
5      <table class="tg">
6        <tr>
7          <th class="tg-hgcj">Address<br><br>[HEX]</th>
8          <th class="tg-hgcj">Master / Slave<br><br>[MD / SD]</th>
9          <th class="tg-amwm">Read / Write<br><br>[R / W]</th>
10         <th class="tg-amwm">Device type<br><br>[--]</th>
11         <th class="tg-amwm">Active<br><br>[TRUE / FALSE]</th>
12         <th class="tg-amwm">Last obtained value<br><br>[FLOAT]</th>
13         <th class="tg-amwm">Config<br><br>[--]</th>
14         <th class="tg-amwm">Is Registrating<br><br>[TRUE / FALSE]</th>
15       </tr>
16 <?php
17 $servername = "localhost";
```

[2] Zobrazené údaje mají pouze reprezentativní charakter, tzn. nejedná se o reprezentaci výsledků práce.

```php
$username = "root";
$password = "";
$dbname = "as01sql";
$conn = mysqli_connect($servername, $username, $password, $dbname);
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
$sql = "SELECT address, ms, rw, devicetype, active, lastvalue, config,
    registrating FROM slavedevtable";
$result = mysqli_query($conn, $sql);
if (mysqli_num_rows($result) > 0) {
    while($row = mysqli_fetch_assoc($result)) {
    echo "<tr>";
        echo "<th class=\"tg-hgcj\">" . $row["address"] . "</th>";
    echo "<th class=\"tg-hgcj\">" . $row["ms"] . "</th>";
    echo "<th class=\"tg-amwm\">" . $row["rw"] . "</th>";
    echo "<th class=\"tg-amwm\">" . $row["devicetype"] . "</th>";
    echo "<th class=\"tg-amwm\">" . $row["active"] . "</th>";
    echo "<th class=\"tg-amwm\">" . $row["lastvalue"] . "</th>";
    echo "<th class=\"tg-amwm\">" . $row["config"] . "</th>";
    echo "<th class=\"tg-amwm\">" . $row["registrating"] . "</th>";
    echo "</tr>";
    }
} else {
    echo "0 results";
}
mysqli_close($conn);
?>
</table></body></html>
```



Address [HEX]	Master / Slave [MD / SD]	Read / Write [R / W]	Device type [--]	Active [TRUE / FALSE]	Last obtained value [FLOAT]	Config [--]	Is Registrating [TRUE / FALSE]
0x00	MASTER	R/W	MASTER/GATE	TRUE	0.0	FFFFFFFF	FALSE
0xAB	SLAVE	READ	SENSOR_FLOAT_R	TRUE	24.254	DF0E235A	FALSE
0x54	SLAVE	R/W	SENSOR_FLOAT_RW	TRUE	11.52	BF7EA523	FALSE
0x21	SLAVE	READ	SENSOR_BOOL_R	FALSE	1.0	874DA412	TRUE
0x11	SLAVE	R/W	SENSOR_BOOL_RW	FALSE	0.0	C78FB2A3	FALSE

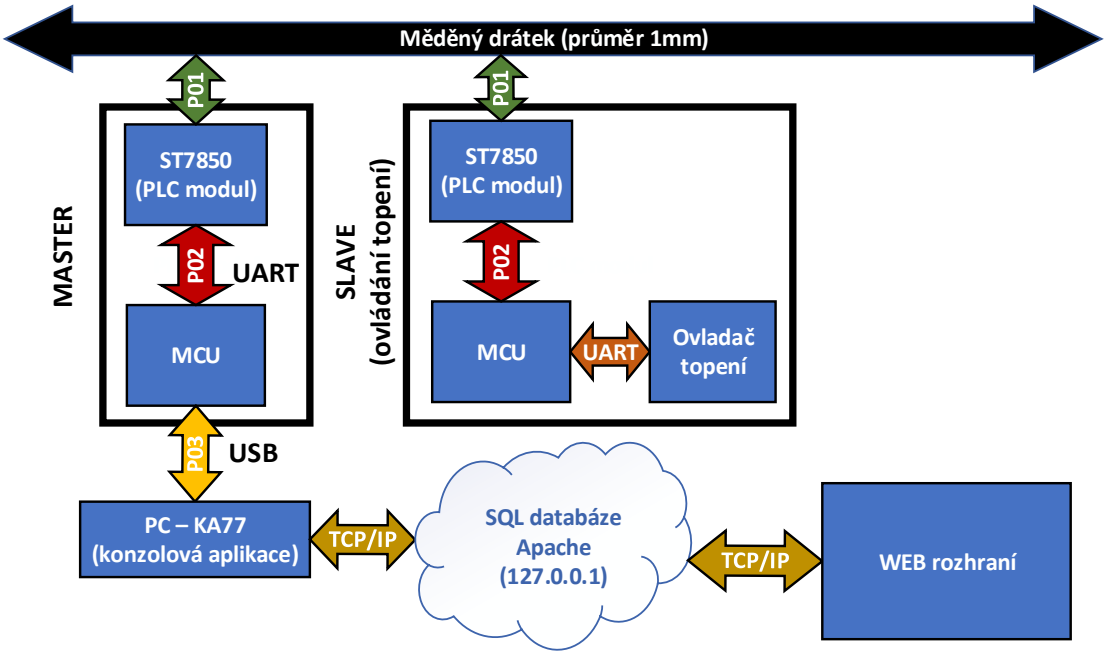
Obr. 25: Ukázka webového výpisu tabulky zařízení.

7 DEMONSTRACE FUNKCIONALITY

Kapitola popisuje fyzické zapojení testovacího pracoviště s ilustračními fotografiemi/schématy a popis několika vytvořených testovacích scénářů. Provedení a výsledky těchto scénářů je rovněž popsáno.

7.1 Popis a ukázka testovacího zapojení

Schéma finálního testovacího zapojení je zobrazeno na obr. č. 26 a následně na obr. č. 27 je vyfotografována praktická realizace.



Obr. 26: Blokové schéma testovacího zapojení.

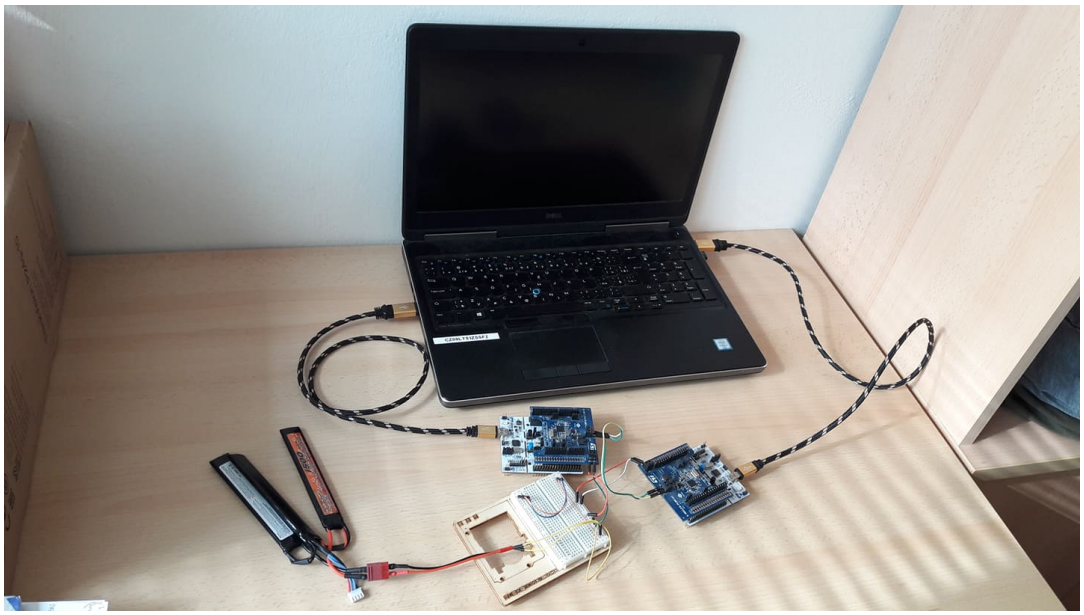
Na zmíněném testovacím BS lze vidět prototyp, se kterým se v průběhu práce pracovalo. Základ tvoří dva koncové PLC moduly, z nichž jeden je MD a druhý SD. Jako přídavný modul v SD byl zvolen ovladač nastavení teploty v domě. Ten je k MCU připojen pomocí UART a přijímá pouze hodnotu typu `float`, která určuje, na jakou teplotu se má dům vytopit. Tato hodnota odpovídá právě hodnotě `LastValue` ve struktuře `Device` (viz podkapitola č. 5.1). Tento typ přídavného modulu byl zvolen z důvodu demonstrace pro zápis i výčet hodnot.

Koncové PLC moduly nejsou připojeny pomocí vazebního členu přímo na silové vedení Zařízení jsou propojeny pomocí měděného drátku. Za účely vývoje a demonstrace není podstatné používat silové vedení, jelikož celý princip komunikace je funkční i za těchto podmínek. Silové vedení by bylo potřeba využít až ve fázi testování např. SNR.

MD je následně připojen k PC, kde je spuštěna KA77 členící přijatá data od MD. Dále je formátuje, zobrazuje a odesílá do databáze umístěné na lokální adrese (127.0.0.1). Odtud

jsou data vyčtena při zobrazení tabulky zařízení pomocí webového rozhraní. Pomocí KA77 se dále i zadávají příkazy pro MD na základě tab. č. 33.

Tato výše popsaná logika je shodná pro všechny následující testovací scénáře a na to navazující měření základních přenosových statistik propustnosti. Od toho je i odvozen výpočet odezvy.



Obr. 27: Fotografie testovacího zapojení.

7.2 Testovací scénáře a výsledky

Za účelem demonstrace funkčnosti bylo vytvořeno několik testovacích scénářů, které zahrnují nejzásadnější aspekty architektury/implementace celé sítě AS01 a jejího monitoringu z NS. Níže jsou všechny scénáře uvedeny a jejich výsledky diskutovány. Scénáře na sebe vzájemně navazují. Veškerá data se opírají o údaje zaznamenané v KA77.

Je nutné správné pochopení celkové funkcionality KA77, které je uvedeno v podkapitole č. 6.2. Zejména se jedná o znalost rozlišení barev výpisů v konzoly, typy zobrazovaných zpráv a finální podoba exportovaných NL rámců.

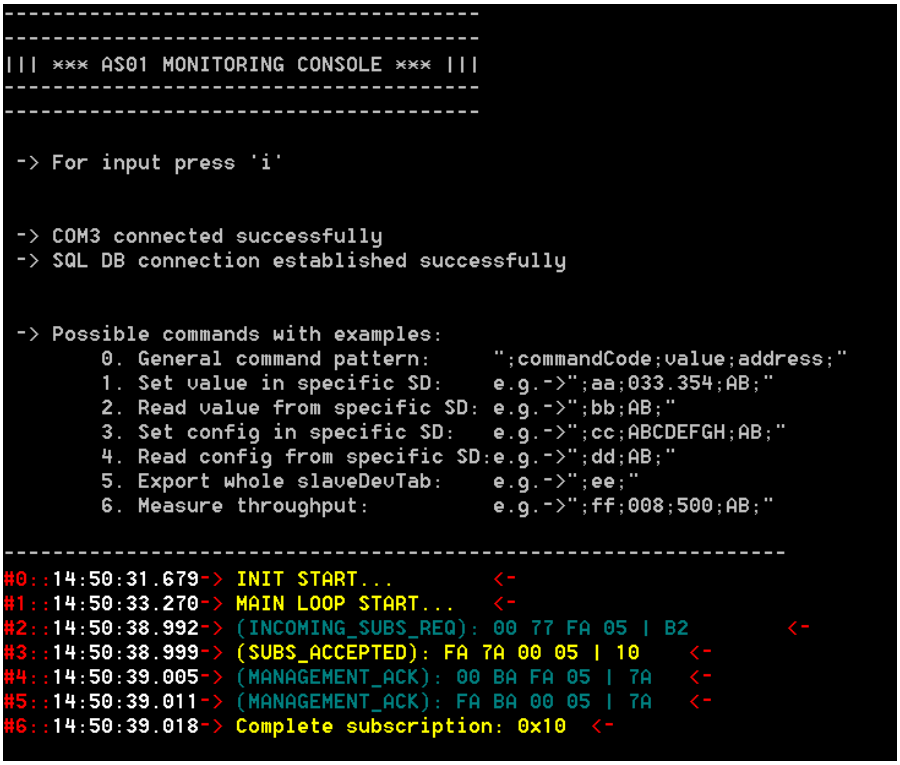
7.2.1 Připojení k napájení

Výpis z KA77 pro tento scénář viz obr. č. 28. Nejprve je spuštěna KA77, kde je vidět, že nejprve dojde k připojení k sériové lince na portu COM3 a po-té následuje test připojení k SQL DB. Slovo test je uvedeno záměrně, jelikož spojení s SQL se otevírá, až jsou k dispozici konkrétní data, která se mají do DB zapsat; následně se pak po dokončení zápisu spojení ukončuje.

Jakmile je KA77 spuštěna a připravena, připojí se MD. To je uvedeno ve výpisu na řádcích `#0` a `#1`. První zmíněný řádek informuje o započaté inicializaci MD. Druhý zase

o připravenosti MD k normální činnosti. Od této chvíle je MD připraveno komunikovat po silovém vedení.

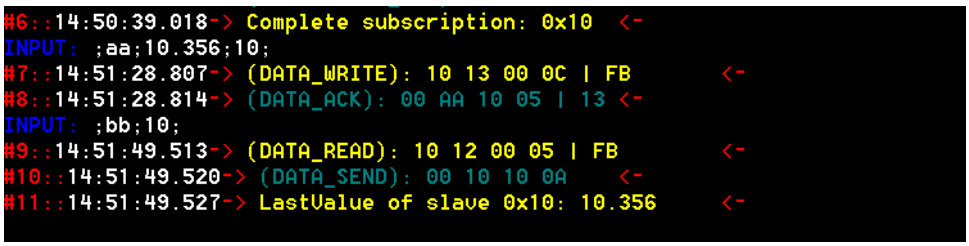
Na zbylých řádcích je zobrazen proces registrace SD do AS01. Tento proces je podrobně popsán v podkapitole č. 4.1.5. Stěžejní je, že registrace SD proběhlo úspěšně a zařízení dostalo přiděleno adresu `0x10` – viz kontrolní výpis na řádku `#6`.



Obr. 28: Snímek výpisu z konzole pro scénář Připojení k napájení.

7.2.2 Výčet a zápis dat ze slave

Tento scénář je zachycen na obr. č. 29. Nejprve je v KA77 stlačena klávesa "i" pro vložení povelu do MD, kde je vložen příkaz[1] pro nastavení proměnné `LastValue` na hodnotu 10.354 v SD s adresou 0x10.



Obr. 29: Snímek výpisu z konzole pro scénář Výčet a zápis dat ze slave.

[1] Všechny příkazy viz podkapitola č. 6.2.1.

Na řádku `#7` je zachycena informace, že MD správně vyslal NL rámec s FTF `DATA_WRITE`. Následuje zobrazení prvních 5 B NL rámce – význam a popis jednotlivých bytů:

1. `0x10` – cílová adresa SD, kde se má proměnná `LastValue` nastavit,
2. `0x13` – hexadecimální kód pro FTF typu `DATA_WRITE`,
3. `0x00` – zdrojová adresa; správně uveden kód pro `MASTER_ADDRESS`,
4. `0x0C` – délka rámce 12 B,
5. `0xFB` – kód pro rozlišení, co SD má s rámcem dělat[2]; může přenášet i jiná data než data k nastavení proměnné `LastValue` (kód s označením `SET_VALUE`)
6. **Hodnota k nastavení** – následuje 7 B, které obsahují samotnou hodnotu k nastavení; tato data se přenáší v textové podobě – konkrétně se jedná o pole typu `char`; nutno podotknout, že se samotná data ve výpisu nezobrazují.

Na řádku `#8` následuje zaslání odpovědi typu `DATA_ACK`, jakožto potvrzení na zprávu `DATA_WRITE`. Toto potvrzení je kladné, lze tedy předpokládat, že SD zařízení svoji hodnotu `LastValue` přenastavil úspěšně. Význam jednotlivých bytů této zprávy:

1. `0x00` – cílová adresa; správně nastaveno na `MASTER_ADDRESS`,
2. `0xAA` – FTF kód pro `DATA_ACK`,
3. `0x10` – zdrojová adresa NL rámce,
4. `0x05` – délka rámce správně 5 B,
5. `0x13` – FTF kód potvrzovaného rámce; správně je uveden kód pro `DATA_WRITE`.

I když SD s adresou 0x10 zaslalo kladné potvrzení, lze si ověřit, zda je to pravda. Vložíme opět povel do KA77 pro výčet `LastValue` ze specifického SD. Nejprve na řádku `#9` je vidět zaslání zprávy typu `DATA_READ`, která je určena právě pro výčet dat ze SD. Jako pátý byte je kód `SET_VALUE`, který značí o žádost výčtu dat `LastValue`. Na to SD správně odpovídá zprávou `DATA_SEND`, kde 5.–11. byte je hodnota `LastValue` převedená na pole typu `char`. Po příjmu zprávy je řádku `#11` MD zařízením vyslán informativní výpis do KA77.

7.2.3 Výčet a nastavení konfigurace přídavného modulu

Data pro tento scénář zachycená v KA77 jsou uvedeny na obr. č. 30. Zde je nejprve mezi řádkem `#11` a `#12` zobrazeno zadání povelu do KA77. Jedná se o povel nastavení nové konfigurace přídavného modulu v příslušném SD.

```
#11::14:51:49.527-> LastValue of slave 0x10: 10.356     <-
INPUT: ;cc;ABCDEFGH;10;
#12::14:52:27.648-> (SLAVE_CONFIG_WRITE): 10 22 00 0C | 41 42 43 44     <-
#13::14:52:27.651-> (DATA_ACK): 00 AA 10 05 | 22        <-
INPUT: ;dd;10;
#14::14:52:46.839-> (SLAVE_CONFIG_READ): 10 21 00 04    <-
#15::14:52:46.843-> (DATA_SEND): 00 10 10 0C | 41 42 43 44      <-
```

Obr. 30: Snímek výpisu z konzole pro scénář Výčet a nastavení konfigurace přídavného modulu.

[2] FTF `DATA_WRITE`

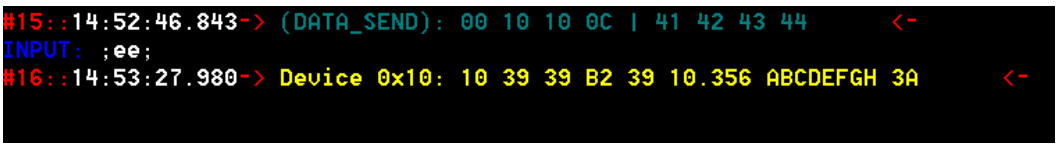
Správně je MD zařízením zaslán rámec typu `SLAVE_CONFIG_WRITE` (řádek `#12`), kde 5. až 12. byte je nová konfigurace v hexadecimální podobě. Na řádku `#13` je kladná odpověď od SD, která značí korektní vykonání rekonfigurace.

Pro ověření nastavené konfigurace je proveden její výčet opět pomocí povelu do konzole. Správně je zaslána zpráva `SLAVE_CONFIG_READ`, na což SD reaguje zprávou `DATA_SEND`, kde v 8 bytovém poli data je obsažena konfigurace přídavného modulu ve SD v hexadecimální podobě.

Nutno podotknout, že zpráva `DATA_SEND` nepotřebuje rozlišit mezi daty `LastValue` a konfigurací, jelikož `LastValue` přenáší 7 B a konfigurace je 8 B. To bylo uvedeno v podkapitole č. 5.1 – `LastValue`, lze nastavit pouze 3 čísla na každé straně od desetinné tečky, zatímco konfigurace má 8 B.

7.2.4 Výčet tabulky slave zařízení z master

Zde je ukázán výpis tabulky zařízení `slaveDevTab` umístěné v MD – viz obr. č. 31. Po zadání povelu je MD vypsán obsah zmíněné tabulky v hexadecimální podobě. Jedná se o tutéž tabulku, která se ukládá do SQL DB, pouze v DB je formátovaná do uživatelsky přívětivější podoby.



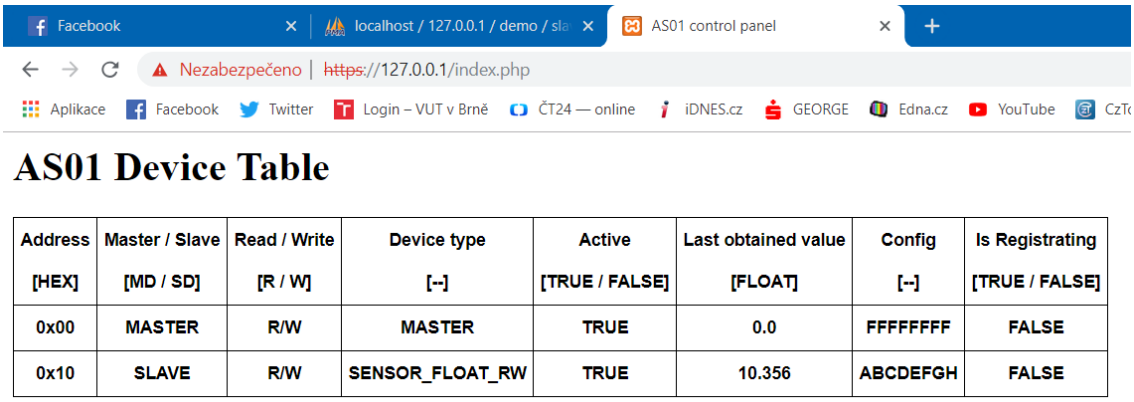
Obr. 31: Snímek výpisu z konzole pro scénář Výčet tabulky slave zařízení z master.

Na výše zmíněném obrázku je tedy korektně uveden výpis tabulky. Jelikož tabulka obsahuje pouze jedno SD s hodnotami nastavenými, tak jak jsou zobrazeny – jedná se o data z předchozím scénářů. Podrobný rozbor jednotlivých zobrazených údajů:

1. **Adresa** – správně uvedena adresa `0x10`,
2. **Master/Slave** – zařízení je slave, takže je správně uveden kód pro hodnotu `SLAVE` (soubor `net_config.h`),
3. **Čtení/Zápis** – správně uvedena hodnota pro `WRITE_DEVICE`,
4. **Typ zařízení** – uvedena hodnota pro `SENSOR_FLOAT_RW`, což je též hodnota nastavená v testovacím SD,
5. **Aktivní zařízení** – zařízení je aktivní, takže je správně hodnota `TRUE`,
6. **LastValue** – hodnota odpovídá údajům nastaveným v scénáři 7.2.2,
7. **Probíhající registrace zařízení** – údaj, zda se zařízení aktuálně nachází v procesu registrace do sítě AS01; toto zařízení je již registrováno, proto se zde správně nachází hodnota `FALSE`.

7.2.5 Zobrazení pomocí webového rozhraní

Zde je pouze zobrazen snímek obrazovky webového výpisu SQL DB – viz obr. č. 32. Tak jak je principiálně uveden v podkapitole č. 6.3.2. Je zde vidět, že data popsaná ve výpisu do konzole KA77 v podkapitole č. 7.2.4 jsou převedena do uživatelsky přívětivější podoby. Navíc oproti zmíněnému výpisu je zde také uvedeno MD zařízení.



Address [HEX]	Master / Slave [MD / SD]	Read / Write [R / W]	Device type [--]	Active [TRUE / FALSE]	Last obtained value [FLOAT]	Config [--]	Is Registrating [TRUE / FALSE]
0x00	MASTER	R/W	MASTER	TRUE	0.0	FFFFFFFF	FALSE
0x10	SLAVE	R/W	SENSOR_FLOAT_RW	TRUE	10.356	ABCDEFGH	FALSE

Obr. 32: Snímek webové stránky s výpisem SQL tabulky zařízení.

7.3 Měření propustnosti a odezvy na NL vrstvě P01

V této podkapitole je provedeno měření propustnosti pro různé velikosti NL rámce, různé velikosti zpoždění hlavní smyčky v MD a následný výpočet odezvy. Tzn. odezva byla vypočítána z dat získaných při měření propustnosti. Nejprve jsou níže uvedeny dvě problematiky, se kterými se v průběhu měření pracuje. Těmito problematikami tedy jsou:

1. **zpoždění hlavní smyčky v master zařízení**,
2. **statická vs dynamická alokace paměti**.

7.3.1 Statická vs dynamická alokace

V průběhu řešení práce se ukázal jako zásadní problém dynamická alokace paměti. Tím je myšlena jak heap tak stack. Jednalo se zejména o problém naznačený níže v bloku kódu:

```
char* tmpCharPointer;
short tmpCharPointerCnt = sprintf(tmpCharPointer, "some info log... \%02X",
    someVar);
debugOutput(tmpCharPointer, tmpCharPointerCnt);

// VS

char tmpCharPointer[32];
short tmpCharPointerCnt = sprintf(tmpCharPointer, "some info log... \%02X",
    someVar);
debugOutput(tmpCharPointer, tmpCharPointerCnt);
```

Zde v první části je použito dynamické pole `tmpCharPointer` typu `char` a ve druhé části má toto pole pevnou velikost 32. Teoreticky by neměl být problém, jak ve správnosti exportovaných dat, jelikož čteme pouze délku dat danou návratovou hodnotou funkce `sprintf`, tak v zasekávání běhu programu. Ovšem v praxi se v průběhu řešení práce zjistilo, že v prvním případě dochází k právě zmíněnému zastavení programu, které se musí řešit tvrdým restartem.

Proto byly veškeré dynamické proměnné odstraněny a nahrazeny právě druhým zmíněným způsobem statickou deklarací. Po této úpravě již k zastavení běhu programu nedocházelo.

V programovém vybavení je tedy použito neblokující přerušení při obluhování UART linek spolu se statickou alokací pamětí, takže není v konečném důsledku již nutné používat zpožďování hlavní smyčky.

Využití v praxi

V reálném nasazení je na statickou vs dynamickou alokaci kladen důraz zejména v systémech s vysokou spolehlivostí, kam spadají např. avionické systémy. Právě pro avionické systémy jsou vydány normy pro psaní SW kódu. Tyto normy např. v Evropské unii zajišťuje Evropský úřad pro civilní letectví (EASA) nebo dále v USA se jedná o FAA (Federální letecký úřad / Federal Aviation Administration).

Co se týče EASA, tak normovaná doporučení pro psaní avionického SW jsou shrnuty v CM-SWCEH-002 publikaci dostupné z [17]. V publikaci na straně 109 lze nalézt informace k problematice statické vs dynamické alokace paměti a problémy pramenící právě z dynamické alokace.

Zpoždění hlavní smyčky v master

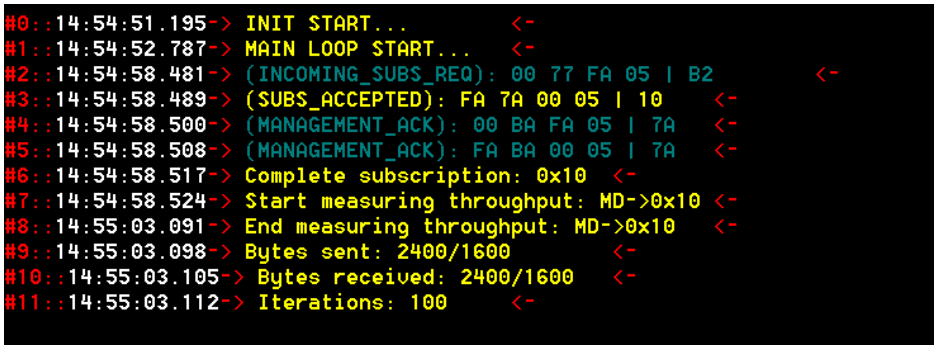
Dokud nebylo přikročeno k řešení statické vs dynamické deklarace, zamezovalo se zaseknutí programu tzv. zpožděním hlavní nekonečné smyčky v MD. Experimentálně bylo zjištěno, že při blokujícím zpoždění alespoň 40 ms na konci každého průchodu hlavní smyčkou je program stabilní. Po použití statické alokace je ovšem program stabilní i bez tohoto zpoždění. Pro porovnání změny hodnot je při měření propustnosti zahrnuto měření i právě se zpožděním hlavní smyčky.

Další zajímavostí je, že při souvislém přenosu rámců (bezprostředně ihned po příjmu potvrzení je odesílán rámec další) je nutných právě oněch zmiňovaných 40 ms. Pokud se, ale nejednalo o souvislou komunikaci, postačovalo i 20 ms.

7.3.2 Metodika měření

Nejprve je nutné vytyčit metodiku měření propustnosti, ze které se v této kapitole vychází. Zapojení zůstává stejné jako je uvedené v podkapitole č. 7.1. Ovšem programové vybavení je upraveno tak, aby se ihned po úspěšné registraci SD spustilo automatizované měření propustnosti.

Příklad měření viz obr. č. 33, kde lze vidět na řádcích `#6` a `#7` začátek automatizovaného měření propustnosti. Po ukončení měření jsou vypsány informace o proběhlém testu. U informací o přijatých a odeslaných bytech je první číslo hodnota velikosti celkových dat NL vrstvy a druhé číslo je hodnota velikosti přenesených dat v poli Data z pohledu NL rámce. Jinými slovy první údaj jsou data počítána i s 4 B hlavičkou NL rámce a druhý bez ní.



Obr. 33: Ukázka výpisu v KA77 pro měření propustnosti.

Nyní je vysvětlena kontinuita komunikace z pohledu návaznosti rámců, přenášených dat a opakování. Výše uvedený text popisuje jedno opakování komunikace, přičemž právě jedno měření je složeno právě ze 100 opakovaných cyklů (iterací). Po dokončení registrace SD, je v MD zahájen proces měření výpisem uvedeným na řádku `#7` výše zmíněného obrázku a započíná se první iterace:

1. Registrace procesu a zaslání zprávy typu `DATA_READ` s aktuální nastavenou velikostí pole Data, pro kterou je aktuální měření určeno; byty pole Data jsou nastaveny na hodnotu `GENERAL_VALUE`.
2. Příjem rámce SD zařízením; zjištění velikosti přijatého rámce, na základě které je provedeno sestavení pole Data rámce pro zaslání odpovědi do MD.
3. Odeslání odpovědi k MD; jedná se o rámec `DATA_SEND` s velikostí pole Data zjištěnou z přijatého rámce.
4. Příjem rámce `DATA_SEND` MD zařízením; proces registrovaný v prvním bodě je vymazán a díky tomu MD ví, že má znova zasílat rámec `DATA_READ`; tzn. začíná se opět od bodu jedna.

Toto měření při 100 iteracích probíhá pro všechny kombinace nastavení uvedené v tab. č. 36. Nutno podotknout, že právě jedna iterace je přenos rámce z MD do SD a zpět z SD do MD; tzn. jedna iterace není jeden přenos jedním směrem.

Tab. 36: Kombinace nastavení pro měření propustnosti.

Parametr	**Hodnoty pro měření**
Velikost pole Data NL rámce	0 B, 8 B, 128 B, 238 B
Modulace	BPSK, 8-PSK
Zpoždění hlavní smyčky MD	0 ms, 1 ms, 40 ms

Maximálně 238 B vs teoreticky možných 251 B jako NL Data Ve výše zmíněné tabulce testovacího nastavení je uvedeno pouze 238 B. To je z důvodu, že byl zjištěn problém s ovladačem použitého pro komunikaci s ST7580. Ovladač nepřijímá rámce větší něž 242 B, tzn. maximální velikost NL Data může být pouze 238 B. Tento problém se nepodařilo v rámci práce vyřešit, jelikož z časových důvodů nebyla provedena hlubší analýza příslušných knihoven daných výrobcem.

7.3.3 Měření propustnosti

Měření je rozděleno na Datový a Režijní kanál. Kombinace měření pro každý kanál jsou uvedeny v tab. č. 36. Níže jsou uvedeny výsledky měření a diskuze výsledků v porovnání s teoretickými hodnotami danými čipem ST7580 uvedenými v podkapitole č. 1.2.2.

Datový kanál

Naměřené výsledky pro Datový kanál jsou uvedeny v tab. č. 37. Jak bylo uvedeno v návrhu sítě AS01, je použita modulace 8-PSK. V uvedené tabulce je nejprve od levého sloupce zobrazeno[3]:

1. **velikost pole Data NL** – velikost pole Data NL rámce v aktuální kombinaci vstupních parametrů při měření,
2. **celkem přeneseno dat** – první údaj s označením Data, je hodnota celkově všech přenesených bytů NL Dat za 100 iterací; údaj s označením Abs. je absolutní hodnota všech přenesených bytů dat za 100 iterací, tzn. i hlavičkou NL rámce,
3. **zpoždění hlavní smyčky** – tři výše deklarované hodnoty zpoždění hlavní smyčky MD (T0 – bez zpoždění, T1 – 1 ms a T40 – 40 ms).

Tab. 37: Měření propustnosti Datového kanálu (8-PSK).

Datový kanál NL vrstva – 8-PSK, 100 iterací								
Velikost pole NL Data [B]	**Celkem přeneseno NL dat [kB]**		**T0**		**T1**		**T40**	
			Doba přenosu [s]	Datová rychlost [B/s]	Doba přenosu [s]	Datová rychlost [B/s]	Doba přenosu [s]	Datová rychlost [B/s]
0	**Data**	0	4,215	-	4,435	-	9,997	-
	Abs.	0,8		189,80		180,38		80,02
8	**Data**	1,6	4,561	350,80	4,811	332,57	12,580	127,19
	Abs.	2,4		526,20		498,86		190,78
128	**Data**	25,6	21,583	1 186,12	21,924	1 167,67	27,059	946,08
	Abs.	26,4		1 223,19		1 204,16		975,65
238	**Data**	47,6	37,274	1 277,03	37,461	1 270,66	41,244	1 154,11
	Abs.	48,8		1 309,22		1 302,69		1 183,20

[3]Tímto způsobem je tab. koncipována i pro režijní kanál a nebude již znovu vysvětlováno její rozložení.

Režijní kanál

Naměřené výsledky pro Režijní kanál, viz tab. č. 37. Zde je použita modulace BPSK, teoreticky tedy bude komunikace probíhat pomaleji než u Datového kanálu.

Tab. 38: Měření propustnosti režijního kanálu (BPSK).

Režijní kanál NL vrstva – BPSK, 100 iterací								
Velikost pole NL Data [B]	**Celkem přeneseno NL dat [kB]**		**T0**		**T1**		**T40**	
			Doba přenosu [s]	Datová rychlost [B/s]	Doba přenosu [s]	Datová rychlost [B/s]	Doba přenosu [s]	Datová rychlost [B/s]
0	**Data**	0	5,203	-	5,418	-	11,858	-
	Abs.	0,8		153,76		147,66		67,47
8	**Data**	1,6	6,381	250,74	6,657	240,35	12,581	127,18
	Abs.	2,4		376,12		360,52		190,76
128	**Data**	25,6	36,726	697,05	37,009	691,72	43,351	590,53
	Abs.	26,4		718,84		713,34		608,98
238	**Data**	47,6	64,487	738,13	64,802	734,55	69,842	681,54
	Abs.	48,8		756,74		753,06		698,72

Interpretace výsledků měření propustnosti

V obou výše uvedených tabulkách je jasně zřetelný rozdíl datových rychlostí pro PSK8 a BPSK. Komunikace přes Datový kanál je pochopitelně rychlejší než přes Režijní, což je žádané. Dále je zde zřetelný také vliv zpoždění hlavní MD smyčky, který má vliv zejména u NL rámců s menší velikostí, kde je markantní pokles datové rychlosti.

Při menší velikosti NL rámců rozdíl není tak markantní jako u větších rámců. To je dáno tím, že samotný přenos rámce zabírá díky své menší velikosti jen část z celkového času přenosu. Tento paradox je číselně vyjádřen a popsán v následující podkapitole.

7.3.4 Číselný rozbor dat měření propustnosti

V této podkapitole je proveden rozbor naměřených časových údajů z měření propustnosti. Tyto naměřené hodnoty se totiž skládají z několika částí, které jsou zde teoreticky spočteny a konfrontovány s reálnými naměřenými daty.

Zapouzdření rámců

Nejprve musí být NL rámec přenesen od MCU k ST7580. Zde je NL rámec zabalen do Lokálního rámce – viz podkapitola č. 4.2.3. Zde komunikace probíhá rychlostí 57 600 bit/s. Data, která mají být odeslána na PLC, jsou zabalena právě do LOC – viz obr. č. 34.

Hlavička Lokální rámec			NL rámec					Kontrolní součet Lokální rámec
STX	*Length*	*CC*	**DAF**	**FTF**	**SAF**	**Lengt**	**Data**	*Kontrolní součet*
1 B	1 B	1 B	4 B až 255 B					2 B

Obr. 34: Praktická ukázka zapouzdření NL rámce pro přenos mezi MCU a ST7580.

A následně na základě informací uvedených v kapitole č. 4 vypadá skladba výsledného rámce přenášeného mezi zařízeními tak, jak je ukázáno na obr. č. 35. Tzn. že NL data vysílaná mezi zařízeními jsou zapouzdřována nejprve do DL rámce a nakonec do PHY rámce. V měření se tedy pracuje pouze s proměnou délkou NL pole Data. Proto je zde provedena analýza jednotlivých časových intervalů, ze kterých se skládá vyslání právě jednoho NL rámce, respektive NL pole Data, která jsou uživatelsky žádaná. Důležitým aspektem při výpočtech je skutečnost, že PHY hlavička se přenáší pomocí BPSK a teprve po příjmu PHY hlavičky přijímač na základě přijatých údajů v hlavičce rozhodne, jak bude probíhat demodulace zbytku rámce – jedná se o pole Mód.

PHY hlavička			DL záhlaví	NL hlavička				NL data	DL CRC
PRE	UW	Mód	*Length*	**DAF**	**FTF**	**SAF**	**LF**	**Data**	*CRC*
4 B	4 B	1 B	1 B	1 B	1 B	1 B	1 B	0–255 B	4 B

Obr. 35: Praktická ukázka zapouzdření NL rámce pro PLC přenos.

Popis vytvořených tabulek

V tab. č. 39 pro Datový kanál a v tab. č. 40 pro kanál režijní je uveden reálný naměřený čas přenosu v porovnání s časem teoreticky potřebným přenosu, tzn. porovnán čas přenosu a čas zpracování dat. V těchto tabulkách je tedy vidět, že čas potřebný pro zpracování dat je několikanásobně vyšší než čas pro přenos. Grafické znázornění závislosti velikosti poměru času potřebného ke zpracování dat v MCU z celkového času potřebného k přenosu na velikost NL rámce viz obr. č. 36 pro Datový kanál a obr. č. 37 pro Režijní kanál.

Konkrétně se ve zmíněných tabulkách provádí rozbor časových intervalů pro jednotlivé úkony části komunikace. Nutno podotknout, že tyto údaje nebyly prakticky měřeny, ale pouze teoreticky vypočítány a konfrontovány s celkovým prakticky naměřeným časem přenosu. Výpis jednotlivých počítaných časů a vzorců, na jejichž základě byl proveden výpočet:

1. $\mathbf{T_{total}}$ – reálně měřený čas přenosu mezi MD a SD; tzn. čas od přijetí zprávy o zahájení měření v KA77 až po čas přijetí zprávy o ukončení měření v KA77.

2. $\mathbf{T_{local}}$ – teoretický čas potřebný k přenosu Lokálního rámce mezi MCU a ST7580

$$\begin{aligned} T_{local} &= \frac{LOC\_frame\,[bit]}{ST7580\_UART\_bit\_rate\,[bit/s]} \cdot 4 \cdot počet\_iterací\,[-] = \\ &= \frac{LOC\_header\,[bit] + NL\_frame\,[bit] + LOC\_check\_sum\,[bit]}{57.600\,bit/s} \cdot 4 \cdot 100 = \\ &= \frac{24\,bit + NL\_header\,[bit] + NL\_Data\,[bit] + 16\,bit}{57.600\,bit/s} \cdot 400 = \\ &= \frac{24\,bit + 32\,bit + NL\_Data\,[bit] + 16\,bit}{57.600\,bit/s} \cdot 400 = \\ &= \frac{72\,bit + NL\_Data\,[bit]}{57.600\,bit/s} \cdot 400. \end{aligned} \tag{4}$$

3. $\mathbf{T_{phy}}$ – teoretický čas potřebný k přenosu PHY rámce mezi MD a SD

$$\begin{aligned} T_{phy} &= \left(\frac{PHY\_header[bit]}{BPSK\_bit\_rate\,[bit/s]} + \frac{DL\_frame\,[bit]}{channel\_bit\_rate\,[bit/s]}\right) \cdot 2 \cdot počet\_iterací\,[-] \\ &= \left(\frac{40\,bit}{9.600\,bit/s} + \frac{DL\_header[bit] + NL\_frame\,[bit] + DL\_CRC\,[bit]}{channel\_bit\_rate\,[bit/s]}\right) \cdot 200 = \\ &= \left(\frac{40\,bit}{9.600\,bit/s} + \frac{8\,bit + NL\_header[bit] + NL\_Data[bit] + 32\,bit}{channel\_bit\_rate\,[bit/s]}\right) \cdot 200 = \\ &= \left(\frac{40\,bit}{9.600\,bit/s} + \frac{8\,bit + 32\,bit + NL\_Data[bit] + 32\,bit}{channel\_bit\_rate\,[bit/s]}\right) \cdot 200 = \\ &= \left(\frac{40\,bit}{9.600\,bit/s} + \frac{72\,bit + NL\_Data\,[bit]}{channel\_bit\_rate\,[bit/s]}\right) \cdot 200. \end{aligned} \tag{5}$$

4. $\mathbf{T_{comp}}$ – jedná se o zpoždění vzniklé zpracováním dat v příslušných MCU a ST7580

$$T_{comp} = T_{total}\,[s] - T_{local}\,[s] - T_{phy}\,[s]. \tag{6}$$

Popis výpočtů číselného rozboru

V této podkapitole je ve zmíněných tabulkách nejprve uveden změřený čas T_{total} přenosu pro 100 iterací měření propustnosti. Následně je vypočítána teoretická doba potřebná pro přenos Lokálních rámců mezi MCU a ST7580 T_{local} za dobu trvání T_{total}. Jedná se o protokol P02, kde komunikace probíhá rychlostí 57 600 bit/s. Vzorec pro výpočet viz vzorec č. 4. V tomto vzorci je číselně proveden rozklad dle obr. č. 34. Nutno zmínit, že ve zmíněném vzorci se objevuje konstanta "4". Tato konstanta znamená, že pro každou právě jednu iteraci je tento LOC přenese mezi MCU a ST7580 celkem čtyřikrát:

1. nejprve při zahájení iterace od MCU (MD) do ST7580 (MD),
2. při příjmu rámce v ST7580 (SD) a jeho exportu od ST7580 (SD) do MCU (SD),
3. při zaslání odpovědi od SD, tzn. od MCU (SD) do ST7580(SD),
4. nakonec export přijatého rámce v ST7580 (MD) do MCU (MD).

Teoretický čas T_{phy} potřebný pro přenos PHY rámce po PLC za použití příslušné modulace a zohlednění výchozí modulace pro PHY hlavičku. Vychází se z zapouzdření NL rámce do DL a následně do PHY rámce (obr. č. 35). Odtud vychází vzorec č. 5, kde lze vidět i právě zohlednění pro jinou modulaci PHY hlavičky. Opět se zde nachází konstanta ("2"), což znamená, že právě při jedné iteraci se průchod přes silové vedení uskuteční právě 2×.

Čas T_{comp} je určen vzorcem č. 6. Jedná o odečtení uvedených teoreticky vypočtených časových údajů od reálně změřeného časového intervalu T_{total}. Posledním sloupcem v probíraných tabulkách je $RATIO_{comp}$. Jedná se o vyjádření, kolik části časového údaje T_{total} zabírá zpracování dat samotným firmwarem v MD/SD, což je právě časový údaj T_{comp}.

Interpretace výsledků číselného rozboru

Dle očekávání jsou teoretické hodnoty pro přenos PHY rámce T_{phy} větší u Režijního kanálu než u Datového. Tomu odpovídá i $RATIO_{comp}$ – celkového času měření T_{total} zabral samotný přenos procentuálně více času než čas strávený zpracováním dat ve firmware.

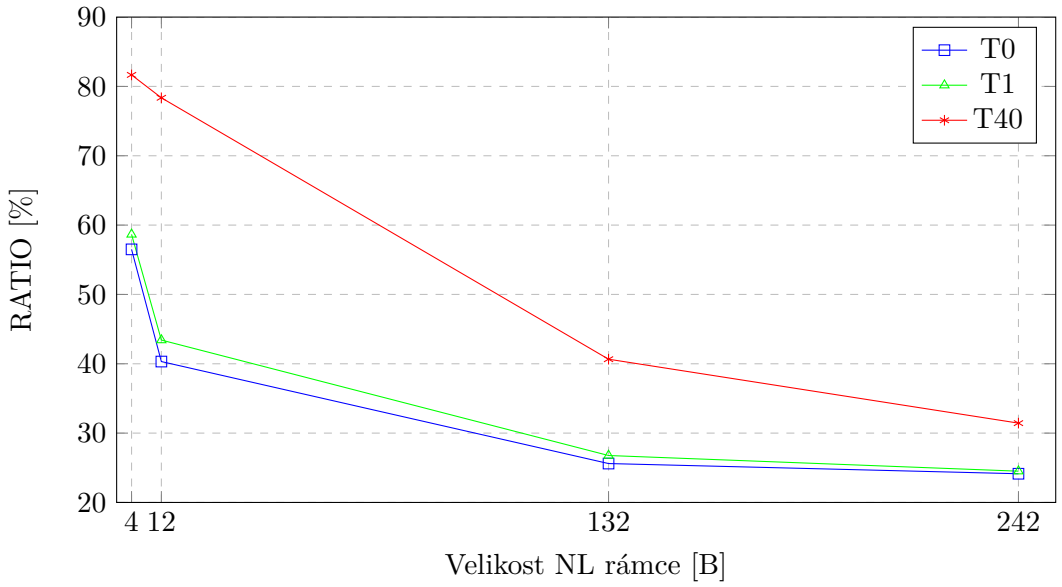
Též lze výsledky považovat za správné z pohledu velikosti NL rámců vs času T_{comp} potřebného pro zpracování dat ve firmware. Jelikož z údaje $RATIO_{comp}$ vyplývá, že poměrově větší část z celkového času přenosu T_{total} je nutná k zpracování dat, pokud jsou NL rámce (samozřejmě analogicky DL/PHY rámce) menší velikosti.

Z pohledu zpoždění hlavní smyčky MD je zřejmé, že při použití větší prodlevy, je čas strávený zpracováním dat firmwarem MD a SD větší než bez zpoždění.

Naměřené, vypočtené a zobrazené hodnoty

Tab. 39: Číselný rozbor naměřených hodnot měření propustnosti pro Datový kanál.

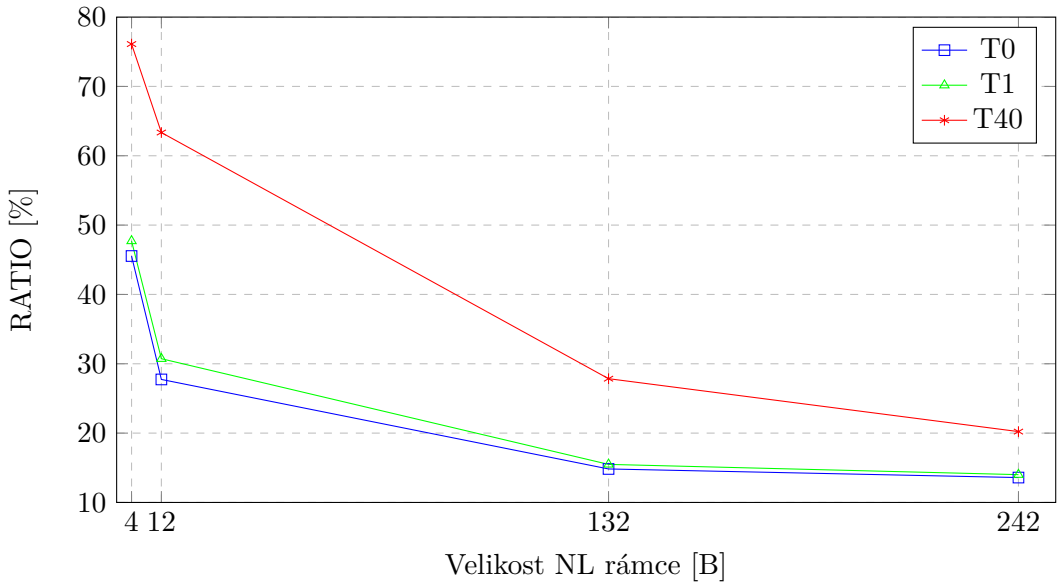
Celkem přeneseno NL/LOC/PHY [B]	**Čas. [ms]**	**T_{total} [ms]**	**T_{local} [ms]**	**T_{phy} [ms]**	**T_{comp} [ms]**	**$RATIO_{comp}$ [%]**
4/9/14	**T0**	4 215	500	1 333	2 382	56,50
	T1	4 435			2 602	58,66
	T40	9 997			8 164	81,66
12/17/22	**T0**	4 561	944	1 778	1 839	40,32
	T1	4 811			2 089	43,42
	T40	12 580			9 858	78,36
132/137/142	**T0**	21 583	7 611	8 444	5 527	25,61
	T1	21 924			5 868	26,77
	T40	27 059			11 003	40,66
242/247/252	**T0**	37 274	13 722	14 556	8 996	24,14
	T1	37 461			9 183	24,51
	T40	41 244			12 966	31,44



Obr. 36: Zobrazení závislosti velikosti NL rámce na poměrném časovém intervalu potřebném pro zpracování rámce v MCU pro Datový kanál.

Tab. 40: Číselný rozbor naměřených hodnot měření propustnosti pro Režijní kanál.

Celkem přeneseno NL/LOC/PHY [B]	**Čas. [ms]**	**T_{total} [ms]**	**T_{local} [ms]**	**T_{phy} [ms]**	**T_{comp} [ms]**	**$RATIO_{comp}$ [%]**
4/9/14	**T0**	5 203	500	2 333	2 370	45,54
	T1	5 418			2 585	47,71
	T40	11 858			9 025	76,11
12/17/22	**T0**	6 381	944	3 667	1 770	27,74
	T1	6 657			2 046	30,73
	T40	12 581			7 970	63,35
132/137/142	**T0**	36 726	7 611	23 667	5 448	14,83
	T1	37 009			5 731	15,49
	T40	43 351			12 073	27,85
242/247/252	**T0**	64 487	13 722	42 000	8 765	13,59
	T1	64 802			9 080	14,01
	T40	69 842			14 120	20,22



Obr. 37: Zobrazení závislosti velikosti NL rámce na poměrném časovém intervalu potřebném pro zpracování rámce v MCU pro Režijní kanál.

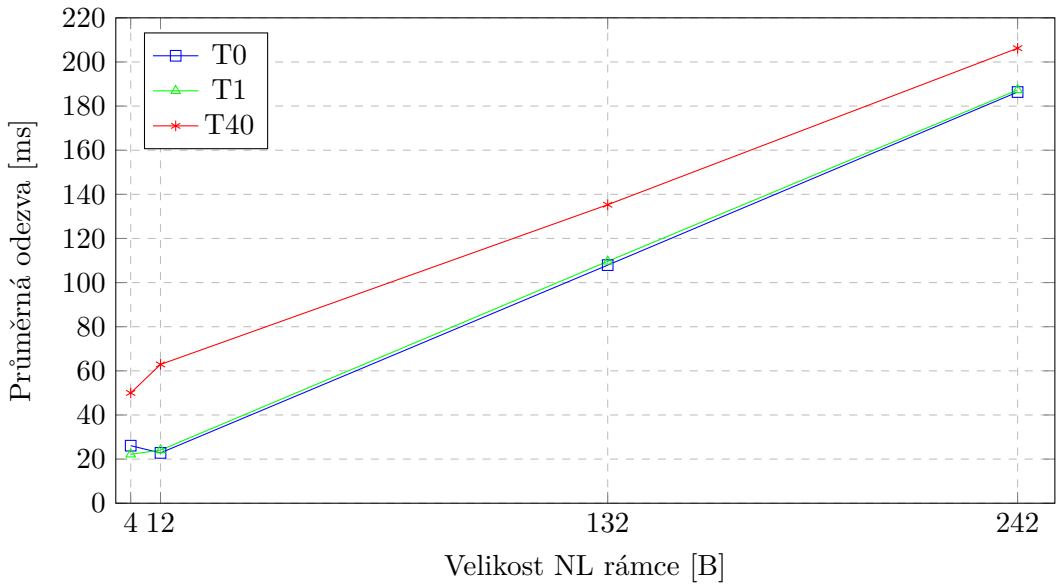
7.3.5 Výpočet odezvy

Souhrn výsledků výpočtů je uvedeno v tabulce č. 41 pro Datový kanál a tabulce č. 42 pro Režijní kanál. Grafické zobrazení závislosti velikosti NL rámce na době odezvy viz obr. č. 7.3.5 pro Datový kanál a obr. č. 7.3.5 pro kanál Režijní. Jak již bylo uvedeno, odezva je vypočítána z dat naměřených při měření propustnosti a to podle vzorce:

$$\frac{doba\_přenosu\,[s]}{celkový\_počet\_iterací\,[-] \times 2} = průměrná\_doba\_jedním\_směrem\,[s]. \tag{7}$$

Tab. 41: Vypočtené hodnoty odezvy pro datový kanál.

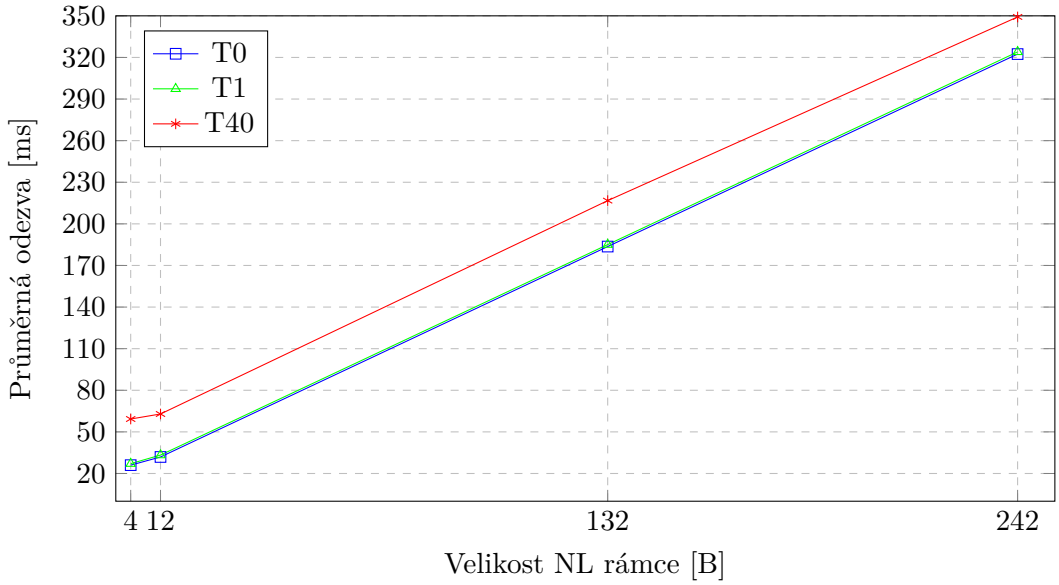
Průměrná odezva - Datový kanál [ms]			
NL rámec [B]	**Zpoždění hlavní smyčky**		
	0 ms	1 ms	40 ms
4	26,075	22,175	49,985
12	22,805	24,055	62,900
132	107,915	109,620	135,295
242	186,370	187,305	206,220



Obr. 38: Zobrazení závislosti velikosti NL rámce na vypočítané době odezvy pro Datový kanál.

Tab. 42: Vypočtené hodnoty odezvy pro režijní kanál.

Průměrná odezva – Režijní kanál [ms]			
NL rámec [B]	**Zpoždění hlavní smyčky**		
	0 ms	1 ms	40 ms
4	26,015	27,090	59,290
12	31,905	33,285	62,905
132	183,630	185,045	216,755
242	322,435	324,010	349,210



Obr. 39: Zobrazení závislosti velikosti NL rámce na vypočítané době odezvy pro Režijní kanál.

Interpretace výsledků výpočtu odezvy

Ve výše uvedených tabulkách je zřejmý rozdíl odezvy pro různé zpoždění hlavní smyčky MD. Dále je dle očekávání odezva vyšší pro přenos datově objemnějších rámců. Takže výsledky se jeví jako reálně možné. Zpoždění má tedy největší vliv u menších rámců, než u větších, protože samotný přenos a zpracování většího rámce trvá delší dobu než rámce menšího; tzn. odezva se lépe ztratí.

8 SHRNUTÍ DOSAŽENÝCH VÝSLEDKŮ

V této kapitole jsou shrnuty dosažené výsledky v rámci vytvořené architektury sítě AS01 včetně diskuze a porovnání s vytyčenými cíli v úvodu práce v kapitole č. 2.3.

8.1 Dosažená funkčnost

V této podkapitole je shrnuta dosažená funkčnost AS01 spolu s NS. Vychází se z dat/výstupů prezentovaných v předchozí kapitole č. 7, respektive testovacích scénářů uvedených v podkapitole č. 7.2. Lze tedy konstatovat, že aktuální stav vývoje sítě AS01 umožňuje:

1. **Registrace SD do sítě AS01**
   - V síti (měděný drátek) je přítomné pouze MD, které je připojeno k NS; toto zařízení je inicializováno a plně připraveno komunikovat.
   - Následně se připojí k síti SD, pak provede inicializaci a zažádá MD o přidělení adresy (registrace do sítě AS01).
   - SD je vyhověno a stává se plnohodnotným účastníkem komunikace.
2. **Nastavení a výčet hodnot z SD**
   - MD umožňuje nastavit hodnotu `LastValue` a konfiguraci přídavného modulu.
   - MD umožňuje vyčíst hodnotu `LastValue` a konfiguraci přídavného modulu.
3. **Komunikace MD s NS**
   - MD formátuje a přeposílá veškerou komunikaci k NS.
   - MD přijímá příkazy k vykonání operací z bodu 2; tím je myšleno i spolu s hodnotami, na které se mají příslušné hodnoty nastavit a ve kterém SD.
4. **Konzolová aplikace KA77**
   - Zaznamenává data přijatá od MD, člení je do tvaru uložitelného v příslušné tabulce SQL databáze a odesílá je do příslušné tabulky SQL databáze.
   - Zobrazuje PLC komunikaci v AS01 díky datům exportovaným z MD.
   - Lze zadávat příkazy k operacím uvedeným v bodě 2.
5. **Webové rozhraní**
   - Zobrazí tabulku zařízení ve webovém prohlížeči na základě SQL DB.
6. **Měření propustnosti sítě**
   - BPSK, 8-PSK.
   - Různé velikosti NL rámců.
   - Různé zpoždění hlavní MD smyčky.
   - Automatizované měření či měření na vyžádání z KA77.

8.2 Diskuze dosažených a vytyčených cílů

Výsledky diplomové práce jsou porovnány s cíli vytyčenými ve výše zmíněné kapitole č. 2.3. Jedná se o sedm hlavních cílů. Nutno podotknout, že jeden z cílů práce bylo i seznámení se s technologií PLC a dodaným testovacím HW.

A) Návrh síťové architektury

Po počáteční analýze technologií byl nejprve proveden rozbor zadání práce. Díky tomuto rozboru byly stanoveny základní požadavky a cíle diplomové práce. Tento rozbor je proveden v kapitole č. 2, respektive v jejích podkapitolách č. 2.1, 2.2 a 2.3.

V první výše zmíněné podkapitole je zadání analyzováno, následuje popis konceptu celkového zapojení a kapitola se uzavírá vytyčením konkrétních cílů práce. Do této chvíle je problematika probírána pouze obecně. Teprve v následující kapitole č. 3 je architektura řešena konkrétně.

Z analýzy zadání práce vyplývá, že požadavkem je vytvoření síťové architektury PLC sítě, která je propojitelná s nadřazeným systém, z něhož je monitorována. Jako příklad aplikace této sítě byl vybrán koncept chytrého domu. Myšlenka je taková, že prvky chytrého domu jsou připojeny na onu síť, díky které je může uživatel monitorovat pomocí různých platforem (např. PC, webové rozhraní či chytrý telefon).

Na základě této hrubé analýzy je vytvořeno hlavní blokové schéma, ze kterého se následně v práci vychází – viz obr. č. 5 – a jsou zde zobrazeny základní prvky celého zapojení. Ze zmíněného obrázku vyplývá, že byla zvolena architektura master-slave, kde master působí jako hlavní článek sítě a zajišťuje spojení s nadřazeným systémem. Zatímco slave reprezentuje právě koncové PLC moduly, které jsou blíže definovány typem přídavného modulu. Tímto modulem jsou zmiňované prvky v chytrém domě (např. teploměr, ovládání topení).

V kapitole č. 3 je na předchozí návrh navázáno a jsou navrženy již konkrétní přenosové parametry navrhované sítě, která byla označena AS01. Jsou nejprve vytyčeny jednotlivé úkoly/pravomoce jednotlivých bloků zapojení v podkapitole č. 3.1. Odtud vyplývá, že master bude v síti obsažen pouze jednou a bude řídit veškerou komunikaci řídit; tzn. slave nebude moci vysílat bez vyzvání mastera.

Po-té jsou určeny přenosové parametry fyzické vrstvy na základě možností, které poskytuje použitý PLC čip ST7580. Byla zvolena možnost duálního kanálu, kde kanál s nižší frekvencí je kanálem datovým a kanál s vyšší frekvencí je kanálem režijním. Od datového se očekává vyšší propustnost, proto byla zvolena modulace 8-PSK. Zatímco u režijního je požadována spíše robustnost vůči rušení, tzn. aby režie v síti probíhala i při větším zarušení silového vedení. Jako modulace byla tedy zvolena BPSK.

Na to v následující podkapitole navazuje řešení metodiky přístupu k médiu. Zde je stanoven princip řešení přístupu pomocí tokenu, který uděluje master. Jsou i vytyčeny případy, kdy může slave bez vyzvání začít vysílat, a to pokud dojde k události, která

potřebuje bezodkladné řešení v master. Příkladem může být aktivace bezpečnostního senzoru.

Dalším řešeným aspektem návrhu AS01 je potvrzování komunikace. Jednoduše řečeno potvrzuje se, až je to bezpodmínečně nutné. Tzn. komunikační protokol PLC sítě by měl být řešen jako stavový.

Poslední podkapitolou je vytyčení způsobu adresace. Adresa byla zvolena velikosti 1 B. To dává teoretický adresní prostor až 256 zařízení. Tento adresní prostor je rozdělen na části a jsou určeny pevné adresy pro jako např. adresa master (0x00) či všesměrové vysílání (0xFF). To je uvedeno v tab. č. 12 popisující rozdělení adresního prostoru.

Zhodnocení Tomuto bodu musela být věnována patřičná pozornost, jelikož se na tomto návrhu dále staví. V průběhu práce se nevyskytla situace, kdy by muselo být přikročeno ke změně nějaké části popsané architektury. Následně se v průběhu práce též nevyskytly nedostatky tohoto řešení, které by byly omezující pro danou aplikaci. Takže lze tento bod považovat za splněný. Celý návrh je vlastním přínosem práce a vycházelo se pouze z myšlenky architektury master-slave a centralizovaného řízení sítě.

B) Návrh P01

Na základě návrhu architektury AS01 se nejprve musel navrhnout protokol P01, který zajišťuje pravidla komunikace po silovém vedení – viz podkapitola č. 4.1. Zde je zejména popsán síťový model (obr. č. 4.1), ze kterého vyplývá zapouzdření jednotlivých vrstev komunikace. Fyzickou a linkovou vrstvu zajišťuje čip ST7580, jejich parametry ovšem lze měnit. Jako data DL vrstvy je právě řešena podoba paketu jako vlastní přínos práce.

Je vytvořena podoba paketu NL vrstvy – viz podkapitola č. 4.1.3. Zde je řešena zmiňovaná síťová adresace, rozdělení na jednotlivé druhy zpráv a skladba přenášených užitečných dat. Tento vytvořený rámec je zapouzdřen do DL rámce. NL rámec je tvořen zdrojovou a cílovou adresou, informací o délce celého rámce a znakem s kódem přenášených dat (FTF). Tyto kódy jsou stěžejní částí NL rámce. Díky nim se konkrétní zařízení rozhoduje k čemu je příslušný NL rámec určen, respektive jaké data jsou v něm přenášena.

Zhodnocení Na základě testování komunikace popsané v kapitole č. 7, tento bod lze též považovat jako úspěšně splněný. Tzn. zprávy v praktické realizaci jsou procesovány přesně tak, jak bylo navrženo. Vlastní přínos práce zde spočívá v návrhu skladby NL rámce a dále v návrhu operací určovaných FTF, které jsou jednotkou určující činnost sítě.

C) Popis P02 a knihovny pro jeho implementaci

Tato část je uvedena v podkapitole č. 4.2. Jedná se pouze o popis způsobu komunikace MCU s ST7580. Zde jsou zejména popsány způsoby ovlivňování parametrů komunikace na úrovni PHY a DL pomocí MIB tabulek (ST7580 je SoC, takže komunikace s ním je předem daná). Tzn. tato část nebyla navrhována, ale byly použity knihovny/drivery dodané od výrobce čipu.

Zhodnocení Zde byl proveden pouze popis funkcí, proto zde nespočívá vlastní přínos. Tento bod lze považovat za splněný, jelikož na základě testování v následujících kapitolách je zřejmé, že komunikace mezi MCU a ST7580 byla pochopena správně.

D) Návrh P03

V této části práce – viz podkapitola č. 4.3 – je navržen protokol pro komunikaci mezi MD a NS. Jde o jednoduchý protokol. Jeho funkce je spíše formální, jelikož v NS lze implementovat pomocí vyšších programovacích jazyků (C# či PHP) jednoduché parsování i většího množství více či méně formátovaných dat.

Je navržena podoba rámce protokolu P03. Skladba rámce neobsahuje adresování, jelikož se jedná pouze a vždy o dvoubodový spoj mezi master a nadřazeným systémem.

Zhodnocení Jak se ukazuje v testovacích scénářích v podkapitole č. 7.2, je tento protokol zcela funkční a lze tedy tento bod považovat za splněný.

E) Firmware pro kontrolér

Na základě návrhů protokolů P01, P03 a popisu P02 je vytvořeno SW řešení pro master i slave zařízení – viz kapitola č. 5.

Nejprve je popsáno řešení uchovávání dat o jednotlivých zařízeních – viz podkapitola č. 5.1. Jedná se o strukturu s názvem Device, ve které je obsažena adresa zařízení, zda se jedná o MD/SD, následně jestli je zařízení pouze pro čtení či i zápis, dalším bytem je kód typu zařízení (např. teploměr, bezpečnostní senzor), zda je zařízení aktivní, jeho nastavená hodnota, jeho konfigurace a jako poslední zda není zařízení v procesu registrace do sítě.

Následující podkapitola se zabývá popisem toku programu – viz podkapitola č. 5.2. Je zde zobrazena logika pro MD i SD. U MD části je nutné věnovat pozornost zejména části vysvětlující procesový přístup, který je vlastním přínosem práce. Jedná se o řešení vytvoření určitých abstraktních procesů, aby MD mělo přehled jakou komunikaci má očekávat. Nejedná se ale o vlákna či více-procesorové řešení.

Zhodnocení Jak se ukazuje v testovacích scénářích v podkapitole č. 7.2 a to zejména v části č. 7.2.1 a 7.2.2, kde je jasně vidět, že MD zachovává kontinuitu komunikace. Proto lze považovat tento bod jako splněný.

F) Návrh a následná implementace připojení k nadřazenému systému

Tato část byla již naznačena v části 2.2. K rozvinutí myšlenky dochází v kapitole č. 6. Zde je rozebráno řešení NS, jakožto konzolové aplikace KA77 na PC, která přeposílá zprávy od MD do SQL databáze, odkud jsou vyčítána z webového rozhraní a PC konzole. Komunikace zde probíhá (když se nepočítá linka USB MD-PC) pomocí standardního TCP/IP modelu.

Konzolová aplikace KA77 je napsána v jazyce C#. Tento vysokoúrovňový jazyk byl zvolen pro snadné řešení tvorby konzolové aplikace. KA77 umožňuje sledovat PLC data, která master vysílá/přijímá. Tato data aplikace člení a odesílá do SQL databáze. Dále umožňuje vkládat příkazy pro ovládání PLC sítě uživatelem. Formát příkazů je předem dán.

Jako ucelený balíček pro řešení databázové a webové části je vybrána distribuce XAMPP obsahující SQL server, Apache server (html server) a PHP funkcionality. Webový server zobrazuje tabulku zařízení uloženou v SQL databázi. Rozvržení této tabulky odpovídá řídící struktuře `Device`.

Zhodnocení Vzhledem k tomu, že webové rozhraní je pouze informativní (nelze zadávat data) a zadávání příkazů přes PC konzoli je neintuitivní, tak tento bod nelze hodnotit jako plně funkční. Původní myšlenka byla tvorba webového GUI. Ovšem zadání práce nespecifikovalo způsob ovládání, proto tento bod lze též považovat za splněný.

G) Vytvoření testovacích scénářů a jejich demonstrace

Tato kapitola s č. 7 je testem celé doposud popsané navržené a implementované funkcionality. Výsledky testování/měření jsou diskutovány v jednotlivých bodech. Ukázka webového výpisu viz bod 7.2.5, kde lze vidět jednoduché webové rozhraní. Část testování zaměřená na funkci komunikace po PLC, tzn. na samotný firmware koncových PLC modulů viz body 7.2.1, 7.2.2, 7.2.3, 7.2.4, je provedena jen částečně. To zejména kvůli velké nabídce testovacích možností a byly tudíž zvoleny pouze 4 základní scénáře. Na výpisech z KA77 je vidět, že kontinuita komunikace je taková, jaká byla navržena.

Měření základních přenosových parametrů propustnosti a odezvy je uvedeno v částech 7.3.5 a 7.3.3. Propustnost byla měřena automaticky a výsledky byly odečteny ve výpisu v KA77. Počáteční a koncový čas měření je časem příjmu příslušné zprávy v KA77. Na základě získaných dat je vypočítána odezva komunikace. Bylo zvoleno několik parametrů pro měření respektive měří se kombinace daných parametrů. Jedná se o různé velikosti NL rámců a zpožďování master hlavní programové smyčky pro datový a režijní kanál.

Na základě výsledků měření lze konstatovat, že propustnost sítě pro Datový kanál se pohybuje v rozmezí cca 190–530 B/s pro síťové rámce s velikostí uživatelských dat 0–8 B. Tato velikost je měřena z důvodu, že vlastní síťová vrstva PLC protokolu vytvořená v této práci ve své specifikaci nepředpokládá rámce větší velikosti. Za účelem porovnání hodnot je provedeno měření i s velikostí 128 B a 238 B, kde je dosaženo rychlosti cca 1220–1310 B/s. Tento výsledek je očekávaný, jelikož přenášet více menších rámců je časově náročnější, co se týče doby zpracování v samotném firmware v MCU. Obdobné měření je provedeno i pro Režijní kanál, kde jsou výsledky 150–380 B/s pro 0–8 B velikost dat, respektive 720–760 B/s pro 128 B a 238 B. Výsledky jsou zde též konfrontovány s různým tzv. časováním hlavní master smyčky, která řešila problém pramenící z nepředvídatelnosti dynamické alokace paměti v MCU.

Změřené výsledky propustnosti jsou číselně rozebrány a konfrontovány s teoreticky vypočítanými hodnotami. Nejprve je popsáno složení reálně přenášených rámců a metodika výpočtu, jelikož pro fázové klíčování je část rámce fyzické vrstvy přenášena jedním druhem zmíněného druhu klíčování a druhá část teprve vlastním zvoleným druhem fázového klíčováním. Zejména byl sledován parametr značící, jak velkou část jednotlivých měření zaujímalo zpoždění způsobené zpracováním dat ve firmware master a slave. Z naměřených údajů vyplývá, že čím menší rámec se po PLC přenáší, tím větší část z celkového času přenosu je částí pro zpracování dat firmwarem. A to cca 45% pro Režijní kanál a 55% pro kanál Datový. Co se týče větších rámců je tato část znatelně menší a to cca 14% pro Režijní kanál a 24% pro Datový kanál.

Zhodnocení Vzhledem k neúplné povedené implementaci nadřazeného systému nebylo možné více otestovat funkčnost z hlediska realizace dalších scénářů. Nicméně vzhledem k vytvořeným měřícím metodikám pro ověření komunikace, které byly realizování a ověřeny základním experimentálním testováním rychlosti a odezvy, lze tento bod rovněž považovat za splněný. To lze podložit konfrontací s teoretickými přenosovými časy na základě parametrů sériových linek a použitých modulací.

9 ZÁVĚR

Na základě zadání práce je navržen komunikační model a příslušné parametry PLC sítě, provedeno seznámení se s technologiemi a zvolenými vývojovými kity STM32F401 (MCU) a X-NUCLEO-PLM01A1 (PLC modem). Celý tento navržený koncept je vytvořen na základě myšlenky aplikace v chytrém domě, jakožto signalizační a monitorovací sítě. Jedná se o úzkopásmovou PLC. Tuto síť lze monitorovat pomocí uživatelských platforem PC (konzolová aplikace) a webového rozhraní.

Jako síťová architektura je zvolena master-slave centralizovaná. Komunikace probíhá dvěma kanály. První je kanál datový, jehož signál je modulován technikou 8-PSK, a druhý kanál je kanál režijní modulovaný technikou BPSK. Datový kanál je určen výhradně pro přenos dat a režijní kanál se stará o režii provozu sítě. Jedná se zejména o registraci nově připojovaných zařízení typu slave do sítě a přidělování vysílacích práv.

Master zařízení reprezentuje bránu k nadřazenému systému a obstarává režii PLC sítě. Master vysílá informační data k nadřazenému systému a přijímá uživatelem přes PC aplikaci zadané povely. Master dále ovládá jednotlivá slave zařízení, zatímco slave zařízení s nadřazeným systémem nekomunikuje. Slave je blíže definován typem použitého přídavného modulu, který do určité míry určuje způsob komunikace.

Na základě návrhu síťové architektury jsou navrženy komunikační protokoly na odpovídajících rozhraních. Je jím protokol P01, který je protokolem na PLC lince, P02 je implementován mezi PLC modemem a řídící jednotkou MCU a P03, který je zamýšlen pouze pro master je protokolem ke komunikaci s nadřazeným systémem. Tyto teoretické návrhy jsou použity jako podklad pro vývoj firmware pro master a slave.

Vznikly dvě verze firmware pro master a slave. Master je pomocí vytvořené konzolové aplikace ovládán z PC. Tato aplikace reprezentuje platformu, přes kterou lze PLC síť monitorovat a ovládat pomocí před-definovaných příkazů. Další částí nadřazeného systému je SQL databáze, do které jsou ukládána přijatá data od master.

Dosažené funkcionality jsou testovány a výsledky jsou prezentovány jak obrazově tak číselně, přičemž reálné číselné výsledky jsou konfrontovány s teoretickými údaji danými výpočtem ze znalosti přenosových parametrů. Toto testování zahrnuje i měření propustnosti a výpočet odezvy.

Nakonec je provedeno shrnutí, diskuze na dosaženými výsledky a identifikace konkrétních nedostatků s možnými způsoby jejich řešení.

LITERATURA

[1] YOUSUF, Muhammad Salman a prof. Mustafa EL-SHAFEI. *Power Line Communications: An Overview - Part I* [online]. 2008 [cit. 2018-12-01]. Department of Systems Engineering King Fahd University ofPetroleum and Minerals, Dhahran, KSA. Dostupné z URL: <https://paginas.fe.up.pt/~ee03130/dissertacao/wp-content/uploads/2011/02/PLC-overview-I.pdf>.

[2] SHARMA, Nutan, Tanuja PANDE a M. SHUKLA. *Survey of Power Line Communication*. Kanpur: International Journal of Computer Applications® (IJCA), 2011, 5s. Dept. of Electronics Engg. H.B. Technological Institute Kanpur, India. Dostupné z URL: <https://pdfs.semanticscholar.org/a0d7/a033fb6aecf429413ed218f1feebb826c977.pdf>.

[3] BANARWAL, Sandeep, Ashish SHARMA, Sukhbani Kaur VIRDI, Himanshu VERMA a Gaurav VERMA. *Power Line Communication*. DEPARTMENT OF ELECTRONICS AND COMMUNICATION, JAYPEE UNIVERSITY. Indian Journal of Science and Technology [online]. Červen 2016, , 4 [cit. 2018-12-04]. DOI: 10.17485/ijst/2016/v9i21/94843. ISSN 0974-5645. Dostupné z URL: <http://www.indjst.org/index.php/indjst/article/download/94843/70106>.

[4] PAVLIDOU, Niovi, A. J. Han VINCK, Javad YAZDANI a Bahram HONATY. *Power Line Communications: State of the Art and Future Trends*. IEEE Communications Magazine [online]. Duben 2003, , 7s [cit. 2018-12-04]. ISSN 0163-6804. Dostupné z URL: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.461.3194&rep=rep1&type=pdf>.

[5] MLÝNEK, Petr. *Analýza a modelování datové komunikace po silnoproudém vedení* [online]. Brno, 2012 [cit. 2018-12-01]. Disertační práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií. Vedoucí práce Doc. Ing. Jiří Mišurec, CSc. Dostupné z URL: <https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=59107>

[6] CANO, Cristina, Alberto PITTOLO, David MALONE, Lutz LAMPE, Andrea M. TONELLO a Anand DABAK. *State-of-the-art in Power Line Communications: from the Applications to the Medium*. [online]. 29.2.2016, 19s [cit. 2018-12-04]. Dostupné z URL: <https://arxiv.org/pdf/1602.09019.pdf>.

[7] CHEN, Shuxian. *Ultra Wideband Gigabit Powerline Communication.* London E1 4NS, United Kingdom, 2009, 203 s. Disertační práce. School of Electronic Engineering and Computer Science Queen Mary University of London. Vedoucí práce Prof. Xiaodong Chen. Dostupné z URL: <https://qmro.qmul.ac.uk/xmlui/bitstream/handle/123456789/442/CHENUltraWideband2010.pdf?sequence=1>.

[8] SADIKU, Matthew N. O., Mahamadou TEMBELY a Sarhan M. MUSA. *High-speed Power Line Communications* Roy G. Perry College of Engineering Prairie View A&M University Prairie View, 2015. ISSN 2248-9622. Dostupné z URL: <http://www.ijera.com/papers/Vol5_issue11/Part%20-%204/M511047072.pdf>.

[9] *UM0932 User manual: ST7580 – FSK, PSK multi-mode power line networking system-on-chip* [online]. 2013, 43s [cit. 2018-12-01]. Dostupné z URL: <https://www.st.com/content/ccc/resource/technical/document/user_manual/8d/ce/5e/34/e0/03/41/ef/CD00270292.pdf/files/CD00270292.pdf/jcr:content/translations/en.CD00270292.pdf>.

[10] STMICROELECTRONICS. *STM32F401RE* [online]. 2018 [cit. 2018-12-01]. Dostupné z URL: <https://www.st.com/content/st_com/en/products/microcontrollers/stm32-32-bit-arm-cortex-mcus/stm32-high-performance-mcus/stm32f4-series/stm32f401/stm32f401re.html>.

[11] STMICROELECTRONICS. *X-NUCLEO-PLM01A1* [online]. 2018 [cit. 2018-12-01]. Dostupné z URL: <https://www.st.com/en/ecosystems/x-nucleo-plm01a1.html>.

[12] *ST7580: FSK, PSK multi-mode power line networking system-on-chip* STMicroelectronic [online]. 2016 [cit. 2018-12-01]. Dostupné z URL: <https://www.st.com/resource/en/datasheet/DM00045903.pdf>.

[13] CARCELLE, Xavier. *Power line communications in practice* [online]. Boston: Artech House, [2009] [cit. 2018-12-05]. Artech House telecommunications library. ISBN 978-1-59693-335-4. Dostupné z URL: <https://books.google.cz/books?id=2cvj-oFvfE0C&pg=PA33&lpg=PA33&dq=master+slave+architecture+academic+plc&source=bl&ots=aH3kOs0rdW&sig=p9axU1LlxYTSXF4Kno0PGWcYe04&hl=cs&sa=X&ved=2ahUKEwig9uv7m4jfAhXRJVAKHX-nDWYQ6AEwDHoECAMQAQ#v=onepage&q=master%20slave%20architecture%20academic%20plc&f=false>.

[14] *Apache HTTP Server Project* [online]. 2019 [cit. 2019-04-07]. Dostupné z URL: <https://httpd.apache.org/>.

[15] *Mysql* [online]. 2019 [cit. 2019-04-07]. Dostupné z URL: <https://www.mysql.com/>.

[16] *Apachefriends* [online]. 2019 [cit. 2019-04-07]. Dostupné z URL: <https://www.apachefriends.org/index.html/>.

[17] CERTIFICATION POLICY AND PLANNING DEPARTMENT, CERTIFICATION DIRECTORATE, EASA. *EASA CM-SWCEH-002*. 09.03. 2012, 111 s. Issue: 01: Revision: 01. Dostupné z URL: <https://www.easa.europa.eu/sites/default/files/dfu/certification-docs-certification-memorandum-EASA-CM-SWCEH-002-Issue\-01-Rev-01-Software-Aspects-of-Certification.pdf>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

ACK	Acknowledgement – Potvrzení bezchybného příjmu
AES	Advanced Encryption Standard – Šifrovací standard
ARM	Advanced RISC Machine – druh CPU
ASK	Amplitude Shift Keying – Klíčování amplitudovým posuvem
AS01	Architektura Sítě 01 – označení pro síťovou architekturu číslo 01
BIO	Basic Input Output – Jednoduchý Vstup a Výstup
BPSK	Binary Phase Shift Keying – dvoustavové fázové klíčování
BS	Blokové schéma
CC	Command Codes – Příkazové kódy
CPU	Central Processing Unit – Centrální procesorová jednotka
CRC	Cyclic Redundancy Check – Cyklický redundantní součet
DAF	Destination Address Field – Pole s adresou destinace
DB	Databáze
DMA	Direct Memory Access – Přímý přístup do paměti
DSP	Digital Signal Processor – Digitální signálový procesor
DL	Data Link layer – Linková vrstva komunikačního modelu
E	Erase – Výmaz určitých dat
FPU	Floating Point Unit – Jednotka zaštiťující operace s čísly s plovoucí desetinou čárkou
FSB	Frame Start Byte – První byte rámce
FSK	Frequency Shift Keying – Klíčování frekvenčním posuvem
FT	Frame Type – Pole typu rámce
FTF	Frame Type Field – Pole s kódem typu rámce
GUI	Graphical User Interface – Grafické uživatelské rozhraní
HAL	Hardware Abstraction Layer – soubor knihoven generovaných programem MxCube
HI	Host Interface – Hostitelské rozhraní

HTML	Hyper Text Markup Language – značkovací jazyk
HW	HardWare – Fyzické vybavení
I/O	Input/Output – Vstup/Výstup
IDE	Integrated Development Environment – Vývojové prostředí
IoT	Internet of Things – Internet věcí
KA77	Konzolová aplikace 77 – Označení příslušné konzolové aplikace
LF	Length Field – Pole délky rámce
LSB	Least Significant Bit – Bit s nejnižší hodnotou v binárním vyjádření
LOC	Local frame – Lokální rámec
MCU	Local frame – Micro Controller Unit
MIB	Management Information Base – Tabulky obsahující konfiguraci čipu ST7580
MSB	Most Significant Bit – Bit s nejvyšší hodnotou v binárním vyjádření
NACK	Not Acknowledgement – Potvrzení příjmu chybného rámce
NRZ	Non-Return To Zero – Metoda kódování bez návratu k nule
NS	Nadřazený Systém
OFDM	Orthogonal Frequency Division Multiplexing – Ortogonální multiplex s frekvenčním dělením
OS	Operating System – Operační systém
PC	Personal Computer – Osobní počítač
PHP	Hypertext Preprocessor – programovací jazyk
PHY	Physical layer – Fyzická vrstva komunikačního modelu
PLC	Power Line Communication – Komunikace po silovém vedení
PNA	Peak Noise Avoidance – Technika vyhýbání se impulzním výkyvům na silovém vedení
PRE	Preamble – Preambule
PSK	Phase Shift Keying – Klíčování fázovým posuvem
P01	Protocol 01 – Protokol s označením 01
P02	Protocol 02 – Protokol s označením 02

P03	Protocol 03 – Protokol s označením 03
QAM	Quadrature Amplitude Modulation – Kvadraturní amplitudová modulace
R	Read – Čtení dat
RISC	Reduced Instruction Set Computer – CPU s redukovanou instrukční sadou
SAF	Source Address Field – Pole zdrojové adresy
SDIO	Secure Digital Input Output – Rozšiřující rozhraní pro SD karty
SDU	Service Data Unit – Protokolová datová jednotka
SFSK	Spread Frequency Shift Keying – Klíčování frekvenčním posuvem rozprostřeného spektra
SNR	Signal to Noise Ratio – Poměr signálu k šumu
SoC	System-on-Chip – Jednočipový počítač
SS	Security Services – Typ rámce/služby na ST7580
SW	SoftWare – Programové vybavení
TACK	Time Acknowledgement – časový interval pro čekání na přijetí ACK v P02
THPE	Time High-Priority-Event – časový interval pro opětovné vyslání příslušného typu zprávy v NL P01
TIC	Time Inter-Character – maximální časový interval mezi jednotlivými znaky v P02
TLT	Time Lease-Token – časový interval po který je platný token v NL P01
TSR	Time Status-Response – časový interval určený k začátku vysílání po přijetí Status zprávy v P02
TTR	Time Token-Response – časový interval pro čekání na přijetí tokenu či potvrzení registrace v NL P01
UART	Universal Asynchronous Receiver and Transmitter – Universální asynchronní přijímač a vysílač
USART	Universal Synchronous / Asynchronous Receiver and Transmitter – Universální synchronní / asynchronní přijímač a vysílač
USB	Universal Serial Bus – Univerzální sériová sběrnice
UW	Unique Word – Unikátní slovo / znak
W	Write – Zápis dat

SEZNAM PŘÍLOH

A PŘILOŽENÉ CD

A.1 Obsah

- CD/
  - Majer_Dominik-ID_164761-DP.pdf
  - src_files/
    - firmware/
      - common/
        - net_config.h
        - PLC_communication.h
        - PLC_communication.c
      - master/
        - master.h
        - master.c
        - masterAPP.h
        - masterAPP.c
      - slave/
        - slave.h
        - slave.c
        - slaveAPP.h
        - slaveAPP.c
    - KA77/
      - Program.cs
      - SQLconnect.cs
    - www/
      - index.php

A.2 Verze nástrojů použitých při vývoji

- **Soubory firmware** – Arm Compiler Version 5.06 update 6
- **Soubory KA77** – .NET Framework 4.6.1
- **Soubory www** – HTML5

B MIB TABULKY

Tab. 43: MIB Konfigurace modemu – 0x00 [9]

Byte	Bit	Název	Popis	Výchozí hodnota
0	0-1	Přístupová PLC vrstva (aktivní vrstva pro příjem PLC rámců)	0: PHY	1
			1: DL	
			2: SS	
	2	Sniffer mód (aktivní DL/SS PLC vrstva)	0: neaktivní	0: neaktivní
			1: aktivní	
	3-4	Délka CRC	0: 1B (8bit)	2: 4B
			1: 2B (16bit, big endian)	
			2: 4B (32bit, little endian)	
			3: 4B (32bit, big endian)	
	5	Rezervováno	Vždy 0	0
	6	Rozsah pro výpočet CRC	0: pouze z DL Dat	0
			1: DL data a pole rámce PHY Length	
	7	Nevyužito	—	0

Tab. 44: MIB PHY data – 0x06 [9]

Byte	Název	Popis	Výchozí hodnota
0-1	Počet přijatých UW polí	Počet PRE a UW polí	0x0000
2-3	Počet validně přijatých PHY rámců	Čítač chybně přijatých DL nebo SS rámců	0x0000
4-5	Počet vyslaných PHY rámců	—	0x0000
6-7	Počet PHY rámců, které byly odmítnuty k vysílání	—	0x0000
8-9	Perioda signálu	Perioda nosné vlny signálu	0x0000

Tab. 45: MIB DL data – 0x07 [9]

Byte	**Název**	**Popis**	**Výchozí hodnota**
0-1	Počet bezchybně přijatých DL/SS rámců	—	0x0000
2-3	Počet nevalidně přijatých DL rámců	—	0x0000
4-5	Počet vyslaných DL rámců	Pouze správně odeslané	0x0000
6-7	Čítač chybně odeslaných DL rámců	—	0x0000

Tab. 46: MIB Konfigurace PHY – 0x01 (první část) [9]

Byte	**Bit**	**Název**	**Popis**	**Výchozí hodnota**
0-2	—	Vyšší frekvence	Zápis frekvence v Hertzích (CELENEC pásmo A, B, C, D)	86kHz (0x014FF0)
3-5	—	Nižší frekvence	Zápis frekvence v Hertzích (CELENEC pásmo A, B, C, D)	72kHz (0x011940)
6	0	Přijímací mód	0: Pouze kanál vyšší frekvence	0
			1: Duální kanál	
	1	Modulace kanálu s vyšší frekvencí	0: FSK	1
			1: Všechny módy PSK	
	2	Modulace kanálu s nižší frekvencí	0: FSK	1
			1: Všechny módy PSK	
	3	Proudová kontrola	0: vypnuta	1
			1: zapnuta	
	4-7	Nevyužito	—	0
7	0-4	Vysílací zesílení	Zesílení PLC datového signálu	21 (0x15)
	5-7	Rezervováno	—	0
8-9	—	ZC prodleva	—	0
10	0-1	Délka PRE pole v PSK rámci	0: 16bit	2
			1: 24bit	
			2: 32bit	
			3: 40bit	
	2-7	Nevyužito	—	0

Tab. 47: MIB Konfigurace PHY – 0x01 (druhá část) [9]

Byte	Bit	Název	Popis	Výchozí hodnota
11	0-1	Datová rychlost FSK modulace	0: 1200bit/s	1
			1: 2400bit/s	
			2: 4800bit/s	
			3: 9600bit/s	
	2	Frekvenční deviace (FSK modulace)	0: 0,5	1
			1: 1	
	3-4	Délka PRE pole FSK modulace	0: 16bit	2
			1: 24bit	
			2: 32bit	
			3: 40bit	
	5	Délka UW pole u FSK rámce	0: 8bit	1
			1: 16bit	
	6	Rezervováno	—	0
	7-8	Nevyužito	—	0
12	—	MSB u UW pole v FSK rámci	Pouze pokud je UW délky 16bitů	0x9B
13	—	LSB u UW pole v FSK rámci	LSB UW pole v FSK rámci	0x58

Tab. 48: MIB SS data – 0x08 [9]

Byte	Název	Popis	Výchozí hodnota
0-1	Počet validně přijatých SS rámců	—	0x0000
2-3	Počet neautentizovaných přijatých rámců	Chybné Digest pole	0x0000
4-5	Počet chybně přijatých SS rámců	—	0x0000
6-7	Počet validně odeslaných SS rámců	—	0x0000
8-9	Počet SS rámců, které byly zamítnuty k vysílání	—	0x0000

Tab. 49: MIB SS klíč – 0x02 [9]

Byte	Název	Popis	Výchozí hodnota
0-15	SS klíč	Klíč pro AES 128bit	0x00000...

Tab. 50: MIB Poslední indikovaná data – 0x04 [9]

Byte	Bit	Název	Popis	Výchozí hodnota
0	0-2	Modulace posledního přijatého rámce	0: BPSK	0
			1: QPSK	
			2: 8-PSK	
			3: BFSK	
			4: BPSK kódovaná	
			5: QPSK kódovaná	
			6: Rezervováno	
			7: BPSK kódovaná s PAN	
	3	Kanál, který přijmul poslední rámec	0: kanál s nižší frekvencí	0
			1: kanál s vyšší frekvencí	
	4-7	Hodnota PGA	PGA hodnota rámce, který byl přijat jako poslední	0
1	—	SNR	SNR počítané z posledního přijatého UW pole (v FSK rámci vždy 0xFF)	0x00
2-3	—	ZC prodleva	Časový interval mezi posledním přijatým bitem UW pole a dobou, kdy proběhne ZC	0x0000

Tab. 51: MIB HI časové intervaly – 0x09 [9]

Byte	Název	Popis	Výchozí hodnota
0	TSR	Hodnota v ms	200 (0xC8)
1	TACK	Hodnota v ms	40 (0x28)
2	TIC	Hodnota v ms	10 (0x0A)

Tab. 52: MIB Verze firmwaru – 0x0A [9]

Byte	Popis	Popis	Výchozí hodnota
0-3	Firmware verze	—	0x00420097

Tab. 53: MIB Informace o poslední potvrzovací zprávě – 0x05 [9]

Byte	**Bit**	**Název**	Popis	**Výchozí hodnota**
0	0-1	Maximální teplota	0: T<70	0
			1: 70<T<100	
			2: 100<T<125	
			3: T>125	
	2-6	Maximální zesílení PLC signálu	Aktivní měření pouze pokud je aktivní proudová kontrola	0
	7	Nevyužito	—	0
1	0-1	Minimální teplota	0: T<70	0
			1: 70<T<100	
			2: 100<T<125	
			3: T>125	
	2-6	Minimální zesílení PLC signálu	Aktivní měření pouze pokud je aktivní proudová kontrola	0
	7	Nevyužito	—	0
2	0-6	Počet nadproudových událostí v průběhu vysílání PHY rámce	Aktivní měření pouze pokud je aktivní proudová kontrola	0
	7	Nadproudové varování (v průběhu posledního přenosu)	0: maximální výstupní proud nedosažen	0
			1: maximální výstupní proud dosažen	
3-4	—	ZC prodleva	—	0